Multi-Objective Genetic Algorithms for the Single Allocation Hub Location Problem

Stephen Yamzuuga Asobiela

Department of Computer Science

Supervisor: Dr. Beatrice Ombuki-Berman

Submitted in partial fulfillment
of the requirements for the degree of

Master of Science

Faculty of Mathematics and Science, Brock University
St. Catharines, Ontario

Contents

List of Tables

List of Figures

Abstract

Hub Location Problems play vital economic roles in transportation and telecommunication networks where goods or people must be efficiently transferred from an origin to a destination point whilst direct origin-destination links are impractical. This work investigates the single allocation hub location problem, and proposes a genetic algorithm (GA) approach for it. The effectiveness of using a single-objective criterion measure for the problem is first explored. Next, a multi-objective GA employing various fitness evaluation strategies such as Pareto ranking, sum of ranks, and weighted sum strategies is presented. The effectiveness of the multi-objective GA is shown by comparison with an Integer Programming strategy, the only other multi-objective approach found in the literature for this problem. Lastly, two new crossover operators are proposed and an empirical study is done using small to large problem instances of the Civil Aeronautics Board (CAB) and Australian Post (AP) data sets.

"I returned, and saw under the sun, that the race is not to the swift, nor the battle to the strong, neither yet bread to the wise, nor yet riches to men of understanding, nor yet favour to men of skill; but time and chance happeneth to them all." Ecclesiastes 9:11. To the God of all creation, I am very grateful for the strength and life throughout this study. You are my all in all.

To my supervisor, Prof. Beatrice Ombuki-Berman, who has been very supportive in every aspect of my study here, I am very grateful. I sincerely appreciate your assistance and excellent supervision that have made this work a success.

Special thanks also go to my supervisory committee members Prof. Brian Ross and Prof. Vladimir Wojcik whose guidance, support, and constructive criticisms have made this work a success.

To Alexander Bailey, I am most grateful for your suggestions, assistance, and proof-reading my work. You were always available to talk to despite your busy schedule. Also, to Michael Medland, who has been more than a friend, I appreciate your impact on my study and life here. To Cale Fairchild, thanks for the technical support you always provided.

Finally, to my family especially my mum, and Mark Akudugu, thank you for the support, encouragement, and guidance throughout my life and education.

Chapter 1

Introduction

Hub location problems occur in real life situations when goods or people are transmitted from one location to another through specialized processing centers. The processing centers direct and facilitate the efficient flow of the data. In these situations, a direct communication between the sender and the recipient is impractical, thus, the need for processing centres. The processing centers are called *hubs*, while the sending and receiving locations are known as *spokes*. Usually, the locations are in a network where a subset of them are chosen to serve as hubs, and the remaining nodes are assigned to each chosen hub. Such a network is termed a *Hub-and-Spoke Network* and form a HLP. The main objective in this network is to efficiently construct a low-cost network that allows for faster transmission of data. This usually involves determining the optimal number of hubs, efficiently locating the hubs, and allocating the non-hub nodes to the hubs. Hub location problems find applications in many areas. Examples of real life applications of HLPs include passenger airlines [5], supply-chain industry [2], telecommunication systems [28], message delivery networks [21], trucking industry [35], among others.

Different variants of HLPs exist such as the single allocation HLPs where a spoke is allocated to exactly one hub [15, 14, 26]; multiple allocation HLPs where a spoke can be assigned to one or more hubs [16, 13]; dynamic allocation HLPs in which a spoke is assigned to a hub when the need arises [10]; *p-hub Median* HLPs where the number of hubs to be chosen is fixed [16, 32, 34], and so on. This thesis investigates both the uncapacitated and the capacitated SAHLP. In the uncapacitated SAHLP (USAHLP), incoming and outgoing traffic flow at hubs are unlimited [6, 25, 32, 36, 22], e.g., the passenger transport system. On the other hand, for the capacitated SAHLP (CSAHLP), the amount of traffic flows at a given hub is restricted to the capacity that hub can handle [33, 29, 12, 17], e.g., Postal Delivery System.

Due to the economic importance HLPs play in real life situations, various research attention has been given to HLPs and a number of heuristics have been proposed in literature to solve them. Some of these methodologies include the simulated annealing (SA) and random descent (RDH) [17]; Lagrangian Relaxation (LR) [11, 4]; ant colony optimization (ACO) [29]; Genetic Algorithms (GA) [33, 34, 24]; integer linear programming formulation [12], among others. Although these methodologies have been able to solve problems instances of nodes sizes $n \leq 200$, their results leave much to be desired in terms of computational time and solution quality.

GAs are search meta-heuristics that model the concept of natural evolution by encoding problem-specific solutions in a chromosome-like data structure and applying the natural process of selection and recombination to evolve approximate solutions for a given problem. GAs have been applied to many areas of research with success. In the SAHLP, GAs have been applied to problem instances of nodes sizes $n \leq 200$ in [33, 34, 24] and obtained modest results. However, none of these GAs has been applied to the newly introduced larger AP problems of node sizes 300 and 400. Besides, there is no previous work on GA using a multi-objective approach to this problem. In order to close these gaps, this thesis proposes two variants of GAs for the problem. The first GA is applied for both the USAHLP and the CSAHLP and uses a single objective criterion measure where the total network transportation cost is minimized as traditionally found in the literature. It employs a list-based chromosome representation [24], and proposes two new problem-specific crossovers that produce efficient results. These new crossovers are the *Best Cost Route Crossover*, inspired by [27, 28], and the *Random Exchange Spoke Crossover*. In addition, the GA employs two other crossovers, thus, the *Double Exchange Cluster Crossover*, and the *Multi-cluster Exchange crossover* from [24, 25] for a comparative study, and incorporates three mutation operators, *Shift Node* [1, 36], *Swap Hub* [1, 36], and *Replace Hub* [24, 25] mutations.

The other GA variant proposes a multi-objective approach to the CSAHLP and it removes the capacity constraints, and instead, introduces the time to process flow as the second objective measure in addition to the network transportation cost as the first objective. This modification to the classical CSAHLP was introduced by [12]. No previous work on GA in the literature has applied a multi-objective approach to this problem. A comparative study for the GA with different multi-objective strategies using Pareto Ranking, Sum of Ranks, and Weighted Sum is done.

Two well-known benchmark data sets are employed to test the performances of the proposed GAs. These are the Civil Aeronautic Board (CAB) and Australian

Post (AP) data sets. The CAB data set is based on air-traffic flows between cities in the USA and its problem instances have node sizes $n = 10, 15, 20$, and 25. It is uncapacitated by nature and hence used for benchmarking the USAHLP. The AP data set on the other hand is based on postal districts in Australia. It has been used as test data for both USAHLP and CSAHLP and includes problem instances of nodes size $n \leq 200$, while new larger instances of node sizes $n = 300$ and 400 have been recently introduced for the USAHLP.

1.1 Summary of Research Motivation and Goals

In a summary, the main objectives of this thesis are:

1. implement a SOGA for the
   - USAHLP capable of solving both small and larger instance data of the AP and CAB Data Sets
   - CSAHLP capable of solving both newer and larger instance data of the AP Data set
2. implement a MOGA for the CSAHLP
3. implement two new crossover operators for the GAs and evaluate their effectiveness, in addition to employing two others from previous work in order to perform a comparative study.

1.2 Organization of Chapters

The remaining chapters are organized as follows. Chapter 2 provides background information on HLPs and a review of past related works, GAs, and multi-objective GA and Fitness Evaluation Strategies. Chapter 3 provides details of the design and implementation of SOGA for SAHLP, while the experimental results of the SOGA for USAHLP and CSAHLP respectively are provided in Chapters 4 and 5. In Chapter 6, the problem formulation and implementation of MOGA for the CSAHLP is provided, while its computational results are shown in Chapter 7. Chapter 8 provides conclusions and future work.

Chapter 2

Background

This chapter provides background information on hub location problems, their classification and formulation, and then a literature review of past research on the problem is made afterwards. Lastly, an overview of genetic algorithms as well as multi-objective genetic algorithms is discussed.

2.1 Hub Location Problems (HLPs)

HLPs are NP-hard [30] combinatorial optimization problems that are concerned with constructing a cost-effective hub-and-spoke network. Constructing a fully connected network that ensures traffic flow from any node can be transmitted to any other node, and which guarantees high-speed connectivity among the nodes is expensive. As such, in situations where a scalable network is required, a *hub-and-spoke* is employed. The overall objective in HLPs, therefore, is to minimize the transportation cost. In a hub-and-spoke network, the connections between hubs form a complete graph, and there is no direct connectivity between any two given spokes thus strictly ensuring that all traffic are routed through the respective hub(s). Given a set of nodes to construct this network, the first step involves *determining* the number of nodes that should serve as hubs, and then the next step involves *locating* these hubs. In the final step, an efficient approach of *allocating* the remaining non-hub nodes (spokes) to the located hubs is done.

Figure 2.1 illustrates a simple hub-and-spoke network. In this figure, hubs are represented as squares, while spokes are represented as circles. As the figure depicts, all the hubs are fully connected. For instance, if node i, which is connected to hub k, is to transmit data to node j which is connected to hub l, it first sends it to its hub (k) and indicates the destination node (j). When k receives it, it checks if j is connected

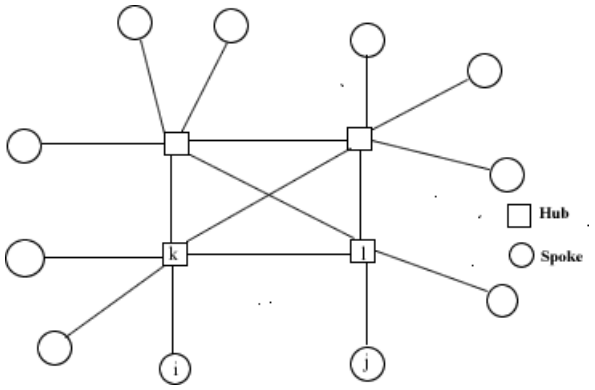

Figure 2.1: Simple Hub-and-Spoke Network

to it or to a different hub. Since j is connected to l, k routes it to l. At l, it checks for the destination node, j, and subsequently sends the data to j. HLPs are classified into two main types. The first type constructs the hub-and-spoke network such that a spoke is assigned to exactly one hub. This type is called SAHLP, and is shown in Figure 2.1. In the second type, called the Multiple Allocation HLP (MAHLP), a spoke can be assigned to more than one hub. Figure 2.2 illustrates a MAHLP.

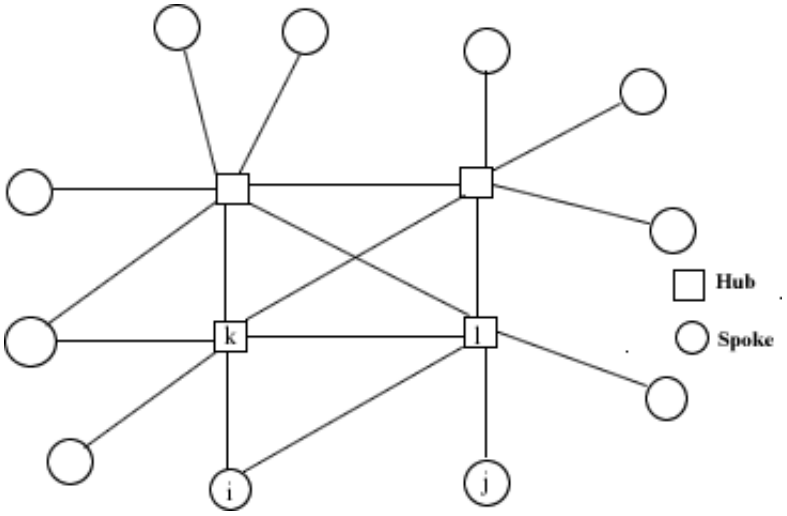

Figure 2.2: MAHLP Network

This thesis concentrates on SAHLP, and in the following section, its formal description is provided.

2.2 Single Allocation Hub Location Problem

As defined earlier, in a SAHLP network, a spoke is assigned to exactly one hub. Its hubs are fully connected as depicted in Figure 2.3, and all inter-nodal flow takes place through at least one hub, and at most two hubs.

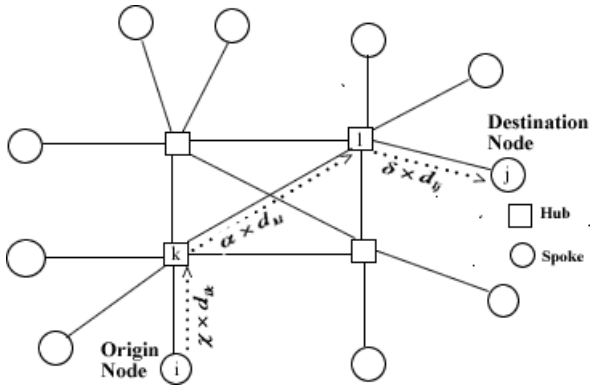

Figure 2.3: SAHLP Network

Two types of SAHLP exist. One type places restrictions on the amount of flow a given hub can receive and process, and is known as the capacitated SAHLP (CSAHLP). A typical application CSAHLP is the postal delivery system (PDS). In a PDS, mail is posted from various source locations (spokes) to destination locations (spokes). First, the mail arrives at a processing center (hub) and is sorted to determine the destination locations. However, any given processing center has a capacity limit within which it can receive and process mail at a given period. As a result, if this capacity is exceeded, any additional mail would have to be redirected to a different processing center whose capacity is not exceeded. In other words, there are capacity restrictions at the hubs beyond which no flow could be accepted and processed. The second type of SAHLP has no restrictions on the amount of flow hubs can receive, and is known as the uncapacitated SAHLP (USAHLP). The passenger transport system is a typical example. No passenger is redirected to another place since there are no limitations on the number of passengers to be serviced.

Irrespective of the type of SAHLP, the main objective is to form a hub-and-spoke network that yields a minimal cost such that every node in the network is either a hub or a spoke and the number of nodes to be chosen as hubs is not known.

2.2.1 Formal Problem Definition

The overall objective for a SAHLP is to find a minimum cost transportation network and this cost is made up of two components.

1. **Internodal flow cost:** the cost of sending data from a source node to a destination node in the hub-spoke network. There are three components of this cost:
   (a) *The Collection Cost (χ)*: the cost of transporting flow from an origin node to its hub (spoke-to-hub flow cost).
   (b) *The Transfer Cost (α)*: the cost incurred in sending flow from the origin node's hub to the destination node's hub (hub-to-hub flow cost), and
   (c) *Distribution Cost (δ)*: the cost of flow from destination node's hub to the destination node. (hub-to-spoke flow cost)
2. **The cost of establishing nodes as hubs.** Depending on the problem, this cost may be fixed for any node to be set up as a hub, or varies depending on the chosen node. For instance, in the CAB data set, the hub establishment cost is fixed irrespective of the node chosen as a hub, whereas with the AP data set, this cost varies with the given nodes.

The above costs are assumed to be per unit distance of flow between the nodes. If node (spoke) i in Figure 2.3 which is connected to hub j, is to transmit flow, W_{ij}, to node (spoke) l connected to hub k, then the internodal flow cost is given as in Equation (2.1). In this equation, χ, α, and δ are the collection, transfer, and distribution costs coefficients respectively.

$$C_{ijkl} = W_{ij}(\chi \times d_{ik} \ + \ \alpha \times d_{kl} \ + \ \delta \times d_{lj}) \tag{2.1}$$

In the formulations that follow, let the flow from node i to node j through hubs k and l be W_{ij}; the number of nodes in the given network be n; the distance from node i to node j be d_{ij}; the cost per unit flow from node i to node j be C_{ij}; the fixed cost of establishing a hub at node k be F_k; $Z_{ik} = 1$ if node i is allocated to the hub located at node k, and zero otherwise; the *transfer* cost be α; the *collection* cost be χ; the *distribution* cost be δ; X_{ijkl} be a decision variable that represents the fraction of flow of traffic from node i to j through hubs k and l; the capacity of hub k be Γ_k. By O'Kelly's Quadratic Integer Programming formulation [26], the cost of an optimal network is given below.

$$\min \sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n}\sum_{j=1}^{n} W_{ij}(\chi \times d_{ik} + \alpha \times d_{kl} + \delta \times d_{lj})X_{ijkl} + \sum_{k=1}^{n} F_k Z_{kk} \tag{2.2}$$

Subject to:

$$\sum_{k=1}^{n}\sum_{l=1}^{n} X_{ijkl} \;=\; 1,\ \forall i,j \in n, \tag{2.3}$$

$$Z_{ik} \;\leq\; Z_{kk}, \forall i,k \in n, \tag{2.4}$$

$$\sum_{j=1}^{n}\sum_{l=1}^{n}(W_{ij}X_{ijkl} \;+\; W_{ji}X_{jilk}) = (O_i + D_i)Z_{ik}, \forall i,k \in n, \tag{2.5}$$

$$\sum_{i=1}^{n} O_i Z_{ik} \leq \Gamma_k Z_{kk},\ \forall i,k \in n, \tag{2.6}$$

$$Z_{ik} \;\in\; \{0,1\},\ \forall i,k \in n, \tag{2.7}$$

$$0 \;\leq\; X_{ijkl} \;\leq\; 1\ \forall i,j,k,l \in n, \tag{2.8}$$

In the above formulations, Constraint (2.3) ensures all the traffic between an origin-destination pair has been routed via the hub sub-network; Constraint (2.4) prevents non-hub nodes from being allocated to other non-hub nodes and Constraints (2.5) and (2.6) restrict the commodity flow through each hub by ensuring their capacities are not exceeded

2.3 Previous Work

A summary of previous work on both USAHLP and CSAHLP is provided here.

2.3.1 Previous work on USAHLP

Abdinnour-Helm [1] proposed a hybrid heuristic called GATS, where a GA determined the number and location of hubs, and a Tabu Search (TS) strategy assigned the spokes to respective hubs. GATS employed a binary encoding representation for the GA part, while *Shift* and *Exchange* moves techniques were used in the TS. According to this work, a *Shift* move is where a spoke is randomly selected from one cluster and shifted

to another randomly selected cluster, while an *exchange* move randomly selects a spoke in a cluster, and makes that spoke a hub of that cluster and the original hub is made a spoke. GATS reported an improvement over results found previously by the same authors using a pure GA which used a distance-based assignment of spokes to hubs; however, the authors have not made the results from their previous GA available.

Abdinnour-Helm and Venkataramanan [3] proposed a new quadratic integer formulation for the USAHLP and employed a branch-and-bound (B&B) procedure and a genetic search in finding near optimal solutions. The B&B method was limited to solving only smaller problems, while the genetic search was efficient in solving large problems.

Topcuoglu *et al.* [36] proposed a GA for this problem and employed a one-point crossover, and introduced two new mutation operators. Their approach was the first to use the AP data set for this problem and reported improved results to some problem instances.

Chen [9] proposed an SA, Tabu List (TL), and improvement procedures to the USAHLP. The approach used SA to determine an upper bound for the number of hubs, and located the hubs using a restricted single allocation exchange approach.

Naeem [24] proposed a GA solution and introduced two chromosome representations as well as three new problem-specific crossovers for the USAHLP.

Silva and Cunha [31] proposed a multi-start TS heuristic and a two-stage integrated TS heuristic to solve this problem. They generated new larger AP problem instances of node sizes 300 and 400 for this problem.

In Filipovic *et al.* [18], a GA-inspired heuristic is proposed. The GA aspect simultaneously finds the number and location of hubs, whiles a B&B procedure is used to allocate spokes to the respective hubs. Their approach also incorporates caching techniques to improve computational performance.

Bailey *et al.* [6] proposed, for the first time, Particle Swarm Optimization (PSO) to this problem, and provided results that compete with those found in literature.

2.3.2 Previous work on CSAHLP

Randall [29] proposed an ACO to this problem and employed four variations of the ACO to explore various construction modelling of the problem as well as incorporated local search heuristics to improve the solutions provided by the ACO approach. The approach, however, was for small size networks only.

In Ernst and Krishnamoorthy [17], a modified version of a Mixed Integer Linear Programming (MILP) formulation developed by them in a previous work for p-hub median problems is proposed. The new formulation used fewer variables and constraints, and they developed two separate heuristic algorithms based on SA and RDH that provide an upper bound. The upper bound was then used to develop a B&B solution method and their results analysis recommended RDH as suitable for small to medium-sized problems, while for larger problems, SA should be used.

In Stanimirovic [33], a GA which employs mutation with frozen bits and a fine-grained tournament selection strategy is proposed. The approach also incorporates a modified one-point crossover and uses caching techniques to improve solutions. It proved efficient in solving large problem instances.

Naeem [24] also proposed a GA solution and introduced two new chromosomes representation as well as three new crossover operators for this problem.

Contreras *et al.* [11] proposed a Lagrangian Relaxation to this problem, and extended their experiments to problem instances of node size $n = 60$, 70, 75, 90, 125, 150 and 175, in addition to the other node sizes of the AP data set.

2.3.3 Previous work on Multi-Objective Approach

Costa *et al.* [12] was the first to propose a multi-objective approach to CSAHLP. They removed capacity constraints on CSAHLP formulation and instead introduced the time to process flow at a hub as the second objective in addition to the transportation cost and proposed an integer linear programming strategy for CSAHLP problem instances of up to 40 nodes.

2.4 Genetic Algorithms (GAs)

GAs [19, 20, 23] are search meta-heuristics that simulate the natural process by which individuals in a given population are randomly selected to reproduce through the use of such genetic operators as crossover and mutation. The individuals selected for reproduction are the fittest in the population. A population in a GA represents the problem search space within which a solution is sought, and each individual in the population is represented by a chromosome which can be translated into a solution in the problem search space. The reproduction process produces new set of individuals that replaces the old population, and this process continues iteratively until a stopping criteria are met, and the solution returned. As a result of their robust nature, GAs

have been successfully applied to many fields of research. In particular, the works of [33, 24, 36, 25, 22, 34] have applied GAs to HLPs.

2.4.1 Simple GA Algorithm

Algorithm 1 outlines a simple GA. It starts by randomly creating a population of individuals, otherwise known as chromosomes, that serve as potential solutions to the problem.

Algorithm 1: Basic GA

```
Set GA parameters
Initialize random population
while termination criteria are not met do
    Evaluate chromosome fitness
    Select chromosomes to reproduce new population
    Perform crossover on selected parents using crossover rate
    Perform mutation on selected offspring using mutation rate
    Replace old population with new population
end
```

2.4.2 Population Initialization

The individuals in the population are initially created here. There are many ways of initializing the population; the commonly used method involves randomly creating the individuals while ensuring that each created individual represents a potential solution in the problem search space.

2.4.3 Fitness Evaluation

In order to determine the quality of each individual in the population, a function that scores the individual in numerical value is employed. Ideally, the function converts the individual into the potential solution of the problem, and then assigns a score to it. In a minimization problem, solutions with smaller fitness scores are considered best, while maximization problems consider higher scores as the best solutions to the problem. This function is called the *fitness function* and its choice for a given problem is mostly dependent on the problem's objective.

2.4.4 Fitness Based Selection

The selection process in GAs is where individuals are chosen for recombination in the form of crossover and/or mutation. The selection is done based on the quality of the individuals. This quality measure, known as fitness evaluation, influences an individual's chances of being selected for recombination: the better the quality, the higher the chance of its selection. This encourages the traits of fitter individuals to be transmitted into the next generations.

There are various types of selection strategies among which are tournament selection, roulette wheel, Boltzman selection, rank selection, steady state selection, and so on. In a tournament selection strategy, a set of k individuals (k is the tournament size) is chosen from the population randomly and the best of these is selected for recombination. Selection pressure is easily adjusted by changing the tournament size. If the tournament size is large, weaker (less fit) individuals have a small chance of being selected. On the other hand, a tournament size of 1 reduces the GA to a random search.

2.4.5 Recombination

Recombination in a GA usually takes the form of crossover where two selected chromosomes undergo reproduction to produce offspring by exchanging genes between them. Different problems require different crossover types to be applied. A simple crossover, called the one-point crossover, randomly chooses a point in a chromosome string, and all genes after this point are exchanged between the chromosomes. This process results in producing two offspring. The new offspring then form part of the new population generated and replace the existing population as evolution continues. Crossovers serve as the main genetic operator in any GA algorithm that are required for evolution.

2.4.6 Mutation

Mutation normally takes the form of a random minor alteration of the genetic composition of a chromosome. Unlike crossovers which involve two chromosomes, during mutation, one chromosome is involved and it produces one offspring after the alteration process. Mutation guarantees that there is diversity in the solutions, and hence minimizes stagnation in the solutions as evolution continues.

2.4.7 Elitism

The concept of elitism is to avoid losing the fittest individual(s) in the population as new generations are evolved by injecting into the new population the fittest individual(s) from the previous population. Such individual(s) are called the elites. Some approaches to elitism exempts these elites from being selected for recombination after copying the elites to the new population, while other approaches allow the elite populate to participate in selection process. In this work, the elite population are allowed to participate in the selection process after they have been copied to the new population.

2.4.8 Termination Criteria

In a GA algorithm, there are three (3) termination criteria. These are

1. The GA has executed for the generation span specified.
2. There is no improvement in the solution quality of the offspring for a certain number of generations.
3. The best solution is obtained.

2.4.9 GA Parameters and Settings

GA parameters and settings form a crucial component in any GA process if good solutions (near optimal) are to be found. Carefully fine-tuning these parameters and settings are catalyst to obtaining better solutions and the vice-versa. Some of these parameters and their settings are explained below.

- *Population Size:* The population size of a GA specifies how many potential solutions (chromosomes) should be kept throughout the evolutionary process. As new chromosomes are created through recombination, the old population is replaced by these new population until a termination criteria is met. A large population size tends to widen the problem search space making convergence to the unique solution to take a longer period, and the vice-versa.
- *Generation Span:* This is the number of generations (iterations) that the GA is set to run.

- *Crossover Rate:* This is the probability of applying a crossover operator during the creation of new individuals in the GA algorithm. Usually two individuals, known as parents, are chosen for this operation to create offspring.

- *Mutation Rate:* The mutation rate specifies the probability of applying mutation in a GA. Since the mutation process involves the random alterations of genes, specifying a higher value for this parameter could reduce the GA to a random search.

- *Elitism Rate:* This is the probability of applying elitism in the GA process. It ensures that fitter individuals are preserved as the old generation is being replaced by the newly created individuals during the evolutionary process.

2.5 Multi-objective Problem

A multi-objective problem has two or more objective measures that determine the overall quality of the solutions produced. The objective measures sometimes do conflict with each other. As a result of these conflicts, optimizing one objective would come at the expense of the other. In order to determine the overall fitness of an individual, some fitness evaluation strategies are mostly employed depending on the problem at hand, and also which strategy best suits the situation. Three of these evaluation strategies are explained shortly.

2.5.1 Weighted Sum

Given a set of objective functions, $f_1, f_2, \ldots, f_k$, of a multi-objective problem, the weighted sum evaluation strategy works by finding a weight parameter for each of the objectives, and then adding them to obtain a single objective function. In Equation 2.9, $w_1, w_2, \ldots, w_k$ are the weight parameters for the objective functions $f_1, f_2, \ldots, f_k$, and *fitness* is the resulting fitness function.

$$fitness = w_1 * f_1 + w_2 * f_2 + \cdots + w_k * f_k \tag{2.9}$$

Finding suitable weights is time-consuming and sometimes biased unless there is prior knowledge of which objective is more important. This is a limitation of this evaluation strategy.

2.5.2 Pareto Ranking

The Pareto ranking strategy was named after an Italian economist Vilfredo Pareto. The idea is to use the concept of dominance to score the individuals in a population using ranks, and replace the raw fitness values with the ranks. Pareto ranking has been applied successfully to multi-objective problems [19].

Based on Veldhuizen and Lamont [37], and considering the fact that MOGA is tackling a minimization problem where smaller values are preferred, the following definition is made:

Given a problem defined by a vector of objectives $\vec{f} = (f_1, ..., f_k)$ subject to appropriate problem constraints, then vector $\vec{u}$ **dominates** $\vec{v}$ iff $\vec{u} \preceq \vec{v}$, $\forall i \in (1, ..., k) : u_i \leq v_i \wedge \exists i \in (1, ..., k) : u_i < v_i$.

From the above definition, a vector is dominated if and only if another vector exists which is better in at least one objective, and at least as good in the remaining objectives

Using this strategy, the raw fitnesses of the individuals are converted into *ranks*, and the appropriate selection criteria is applied using these ranks. The ranking of the individuals is done by first going through the population to find individuals that are non-dominated, and assigning a rank of 1 as their score. After being ranked, these set of individuals are removed from the unranked population. In the second round of iteration, the remaining unranked population is again checked for non-dominated individuals, and the resulting individuals given rank 2, and subsequently removed from the unranked population.

The process continues with each step's non-dominated individuals each given a rank of *previous_rank + 1* until the entire population has been ranked. Individuals assigned rank 1 are non-dominated (best); $i + 1$ dominated by those of rank i, and so on. The effect of the ranking is that, it stratifies the population into categories. In order to know which individual in the list of rank 1 individuals is better than the other, the raw fitness values have to be resorted to.

Algorithm 2 outlines how a given population is ranked using the Pareto Ranking Scheme.

2.5.3 Sum of Ranks

The sum of ranks is another fitness evaluation technique employed for most multi-objective problems. The strategy was proposed by Bentley and Wakefield [7]. The motivation for its conception was an attempt to solve the limitations that higher di-

Algorithm 2: Pareto Ranking Scheme Algorithm [27]

```
current_rank ← 1
N ← pop_size
m ← N
while m ≥ 1 do
    for i ← 1 to m do
        if v_i is non-dominated then
            rank(v_i) ← current_rank
        end
    end
    for j ← 1 to m do
        if rank(v_j) = current_rank then
            Remove v_j from population
            N ← N − 1
        end
    end
    current_rank ← current_rank + 1
    m = N
end
```

mensional multi-objective problems face in which Pareto Ranking struggles to provide good results. It has, however, been successfully applied to small-dimension problems as well by Bergen and Ross [8]. One major advantage of this strategy is that, it eliminates outliers in the solutions [8].

For a given set of fitness vectors of an individual, the approach works as follows. It first separately ranks each of the objectives by assigning rank 1 to individuals with the best fitness score of that objective, rank 2 to the next best individuals, until every individual has been ranked. This process is repeated for the rest of the objective measures. After all the objectives have been ranked, the sum of all the ranks is computed, and this gives the fitness score of the individual.

Two types of this approach exist. The first type works as explained above where the raw ranks are summed to produce the overall score. The second type, called the normalized sum of ranks, divides each of the ranks of a given objective by the sum of all the ranks in that objective. This thesis uses the normalized sum of ranks since it guarantees that the ranks are fairly and evenly distributed throughout the entire population.

The resulting summed ranks then replaces the raw fitnesses of the individuals and an appropriate reproduction selection strategy is applied using these ranks.

One major difference between the sum of ranks and weighted sum is that, the

weighted sum adds the raw fitness scores of the various objectives of the individuals after finding the weight parameters, unlike sum of ranks which first ranks the given objective measures, before summing the ranks together.

Chapter 3

Single Objective Genetic Algorithm for the Single Allocation Hub Location Problem

This chapter discusses the design and implementation of the SOGA for SAHLP, the chromosome representation and initialization, fitness evaluation, the selection strategy, crossovers, and mutation.

SOGA starts by randomly initializing the population's chromosomes where every node has a potential of being made a hub. Each chromosome in the population is transformed into a network of one or more clusters. After evaluating the fitnesses of the chromosomes, evolution starts where a fitness-based selection strategy that employs tournament selection with elite retaining model [19] is used for selecting chromosomes for reproduction. Two new problem-specific crossover operators have been proposed in addition to employing two others from previous works [24]. The evolutionary process continues iteratively until a stopping criteria is met, at which stage, SOGA stops and returns the best chromosome. Algorithm 3 outlines the pseudocode for SOGA.

Algorithm 3: SOGA for the SAHLP

```
Read problem instance data set
Set GA parameters
Randomly create initial population
while termination criteria are not met do
    Evaluate fitnesses of individuals using Algorithm 4
    Add elite individuals to new population using elitism rate
    while the size of new population < population size do
        Select chromosomes using tournament selection strategy
        Perform crossover using crossover rate
        Perform mutation using mutation rate
        Add offspring to new population
    end
    Replace old population with new population
end
Return the best individual
```

3.1 Chromosome Representation and Initialization

A direct chromosome representation, known as the list-based representation [24], was employed for encoding the chromosome. This representation scheme uses integers in the range $1...n$ to represent the nodes in the given network, where n is the number of nodes in the network. Here, a chromosome is represented as an array with length equal to n. The indices of the array represent each of the nodes, and the array entries are the hubs which the given nodes have been assigned to, while a hub is assigned to itself in the array. For instance, if node k is a hub, then k is assigned to array index k.

Figure 3.1 shows how a given network is encoded using the list-based representation. In this figure, there are 10 nodes, of which three are selected as hubs. The hubs are 3, 4, and 6. As can be observed, nodes 5 and 7 are assigned to hub 6 forming one cluster. A cluster in the context of this thesis is a group of nodes, including the hub itself, that are assigned to a hub. In transforming this cluster to the list-based representation, 6 (the hub) is the entrant to array indices 5 and 7, and since a hub is assigned to itself, 6 is assigned to array index 6. The transformation process is shown by the arrow in the diagram. The same transformation process is repeated for the other two clusters in the network where nodes 10, 1 and 2, allocated to hub 4, have 4 as the entrant to their respective array indices, i.e, 10, 4, 1, and 2, and nodes 8 and

9 allocated to hub 3.

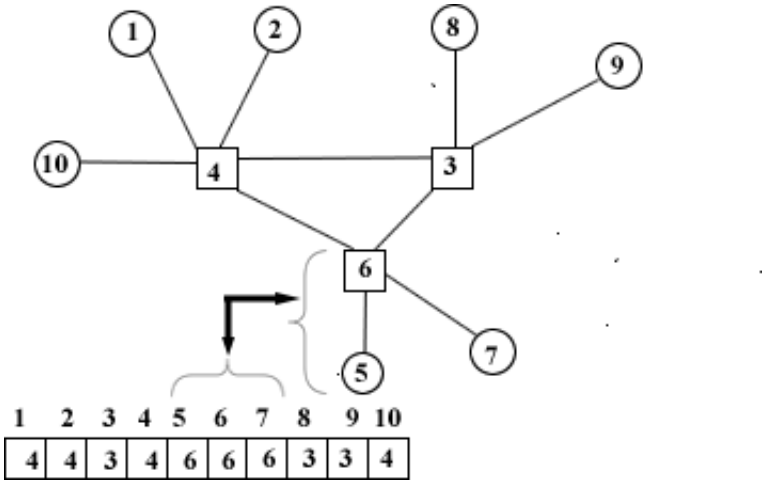

Figure 3.1: List-based Chromosome Representation

During chromosomes initialization, three (3) main steps were followed. These are *determination* of the number of potential hubs, *location* of these potential hubs, and *optimal allocation* of spokes to the located hubs. In determining the number of hubs to be used in the network, m, it is randomly selected in the range $2 \leq m \leq n/2$. After determining the number of potential hubs, the next step is to locate these hubs. In our approach, any node has an equal potential of being made a hub. Therefore the hubs were randomly chosen from the set of nodes until the required number of hubs is met. The distance-based assignment of spokes rule was used in the the *allocation* of spokes step. This rule ensures spokes are assigned to hubs that are closer to them using their computed Euclidean distances. This process is repeated until all spokes are assigned the chosen hubs. The final stage in the initialization process is a *repair* step which re-assigns (demotes) *stand-alone* hubs to other hubs. Stand-alone hubs are hubs that do not have any spoke assigned to them. In the case of CSAHLP, the capacities of the hubs are checked, and if there is an excess flow in any hub, the assigned spokes are re-assigned elsewhere until no excess flow exists. The above initialization process is repeated until the required population size is obtained.

3.2 Fitness Evaluation

The weighted-sum fitness function was used and it computes the transportation cost of flow through the entire network as shown in Equation 3.1. In this equation, the cost of the internodal flow is the first term, while the second term denotes the hub

establishment cost.

$$f(x) = \sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n}\sum_{j=1}^{n} W_{ij}(\chi \times d_{ik} + \alpha \times d_{kl} + \delta \times d_{lj})X_{ijkl} + \sum_{k=1}^{n} F_k Z_{kk} \tag{3.1}$$

Algorithm 4 shows how the computation of the fitness function in Equation 3.1 is done. In this algorithm, n is the number of nodes in the network; W_{ij} represents the flow from node i to node j through hubs h_i and h_j respectively; $H_c(hub)$ is the hub establishment cost of node *hub*; χ, δ, and α are respectively the collection, distribution and transfer cost of the network; *hubList* is the list of hubs in the given chromosome.

Algorithm 4: Computing the fitness of a chromosome

Input: chromosome
Output: fitness
$fitness \leftarrow 0$;
for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow 1$ **to** n **do**
    $fitness \leftarrow fitness + W_{ij} * (\chi * d_{i,h_i} + \alpha * d_{h_i,h_j} + \delta * d_{h_i,j})$;
  end
end
foreach *hub in hubList* **do**
  $fitness \leftarrow fitness + H_c(hub)$;
end
return $fitness$;

3.3 Recombination

The recombination operators employed are explained next.

3.3.1 Best Cost Route Crossover (BCRC)

The BCRC was inspired by the works of [27, 28] in Vehicle Routing Problems with Time Windows. In these works, BCRC randomly chooses a route from each parent,

and inserts the customers from the chosen routes at locations in the opposite child that yields minimal (optimum) cost. This thesis modified BCRC for SAHLP and it involves two main steps:

- random selection of a cluster from each parent, and
- optimal allocation of the nodes in the selected clusters at places which results in minimal cost in the opposite children.

During the cluster selection stage, random hubs, $h1$ and $h2$, are chosen from both parents, $p1$ and $p2$ respectively, and marked for exchange and their assigned spokes are added to a re-assignment list. Hub $h2$ is then copied to child $c1$ if it is not an existing hub in $p1$. The same operation is done for hub $h1$ to child $c2$.

The next stage is to allocate the nodes in the re-assignment list to cost-efficient resulting clusters in the offspring. During this process, the spokes are randomly selected from the re-assignment list, and the cost of allocating the selected spoke to any cluster is computed. In computing this cost, a node from each cluster in the child is randomly selected, and the transportation cost to and from these nodes is computed for each possible allocation of the randomly selected node in the re-assignment list. After estimating the optimal cost, the selected node is assigned to the hub of the node which yielded the lowest cost. This process is repeated until all unallocated nodes are assigned.

Algorithm 5 shows the pseudo-code for BCRC procedure and an illustration of how two chromosomes undergo BCRC is shown in Figure 3.2. In Equation 3.2, the function for computing the optimal cost in order assign a given node in the BCRC is given. In this equation, i is the node to be assigned, while N_s is the set of nodes made up of each randomly selected node from each of the clusters in the child solution. The function computes the cost of transporting flow from the randomly chosen node in the reassignment list to and from each node in N_s.

$$C(x) = \sum_{j \in N_s} W_{ij}(\chi * d_{ik} + \alpha * d_{kl} + \delta * d_{lj}) + W_{ji}(\chi * d_{jl} + \alpha * d_{lk} + \delta * d_{ki}) \quad (3.2)$$

Algorithm 5: The BCRC Algorithm

Input: Two parents

Output: Two children

foreach *parent in Parents* **do**
Randomly select a cluster
end
Copy parents to children excluding selected cluster from opposite parent
foreach *child in children* **do**
while *there are nodes in selected cluster from opposite parent* **do**
Randomly detach a node, *randomNode*
Randomly select a node each from the clusters of *child*, and add to N_s
Compute the cost of transporting flow from *randomNode* to and from every node in N_s using Equation 3.2
Allocate *randomNode* to the hub of the node that yielded the minimum cost from the above step of computation
end
if *CSAHLP* **then**
foreach *hub in Hubs* **do**
Check its capacity and reassign spokes to other hubs if exceeded
end
foreach *hub in Hubs* **do**
Check if it is standalone and demote if so
end
end

3.3.2 Random Exchange Spokes Crossover (RESC)

The RESC operator is proposed in this work. It randomly selects a cluster from each of the mating parents, and swaps the spokes from the selected clusters. During the swapping process, the spokes are randomly detached from the selected clusters, and then allocated to the opposite child's selected cluster until there are no more spokes in the selected clusters. Missing nodes are allocated, and standalone hubs are demoted.

To illustrate how RESC works, consider two parents, P_1 and P_2 shown in Figure 3.3, selected for crossover, with $P_1 = \{\{\underline{2}, 3, 4\}, \{\underline{5}, 1, 6\}\}$ and $P_2 = \{\{\underline{3}, 1, 2, 4\}, \{\underline{5}, 6\}\}$. The underlined nodes in these representation are the hubs for the respective clusters. RESC is applied as follows.

1. Randomly select a cluster from P_1, say, $P_1C_1 = \{\underline{2}, 3, 4\}$, and a cluster from

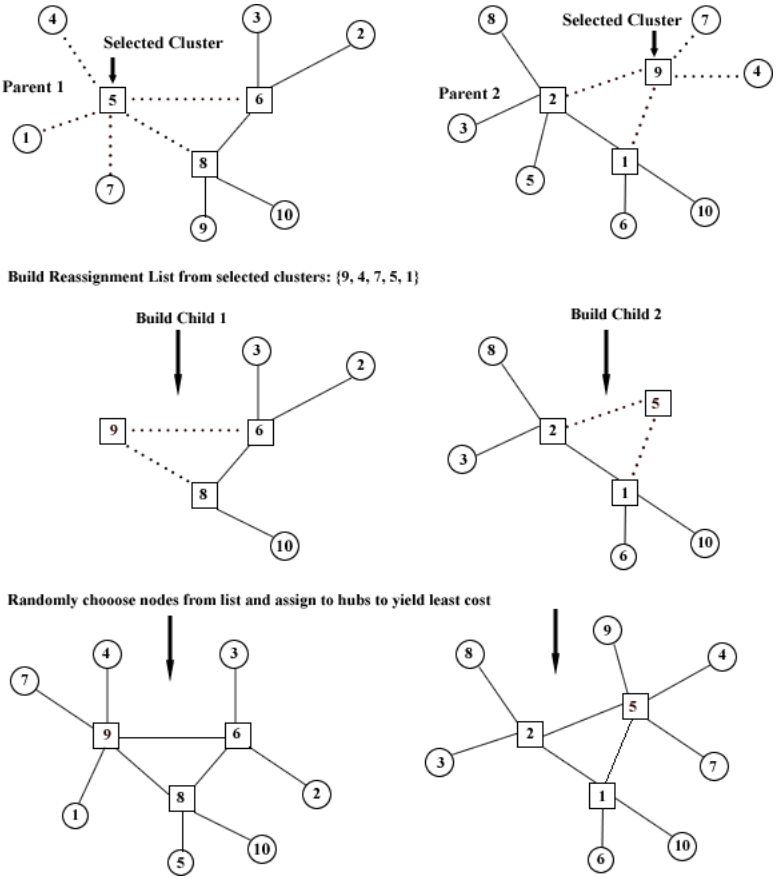

Figure 3.2: Illustration of the BCRC Operator

P_2, say, $P_2C_2 = \{\underline{5}, 6\}$.

2. Detach the spokes in P_1C_1 and add to a "to-be-assigned" nodes list, L_1, and repeat the same process for P_2C_2, L_2. Thus, $L_1 = \{3, 4\}$, and $L_2 = \{6\}$.

3. Detach from P_1, spokes that are in P_2C_2, and add to L_1. If a spoke from P_2C_2 is a hub in P_1, it is not detached. The same process is repeated for P_2. Duplicate nodes in either lists are deleted as well as nodes that are hubs in destination child. This step results in $L_1 = \{3, 4, 6\}$, and $L_2 = \{6, 4\}$.

4. Copy the genetic information of the P_1 to its child, Ch_1, and repeat this for P_2 and its child, Ch_2 and this results $Ch_1 = \{\{\underline{2}\}, \{\underline{5}, 1\}\}$, and $Ch_2 = \{\{\underline{3}, 1, 2\}, \{\underline{5}\}\}$.

5. Randomly allocate the nodes in L_1 to P_1C_1 in Ch_1, and then repeat this process using L_2 for P_2C_2 in Ch_2.

6. For a CSAHLP, where a node to be allocated in step 5 will result in excess

flow in the selected cluster's hub, it is allocated to any other hub that yields an optimal cost. Standalone hubs are demoted if any exists. The created children are: $Ch_1 = \{\{\underline{2}, 4, 3, 6\}, \{\underline{5}, 1\}\}$, and $Ch_2 = \{\{\underline{3}, 1, 2\}, \{\underline{5}, 6, 4\}\}$.

Figure 3.3 illustrates the above processes.

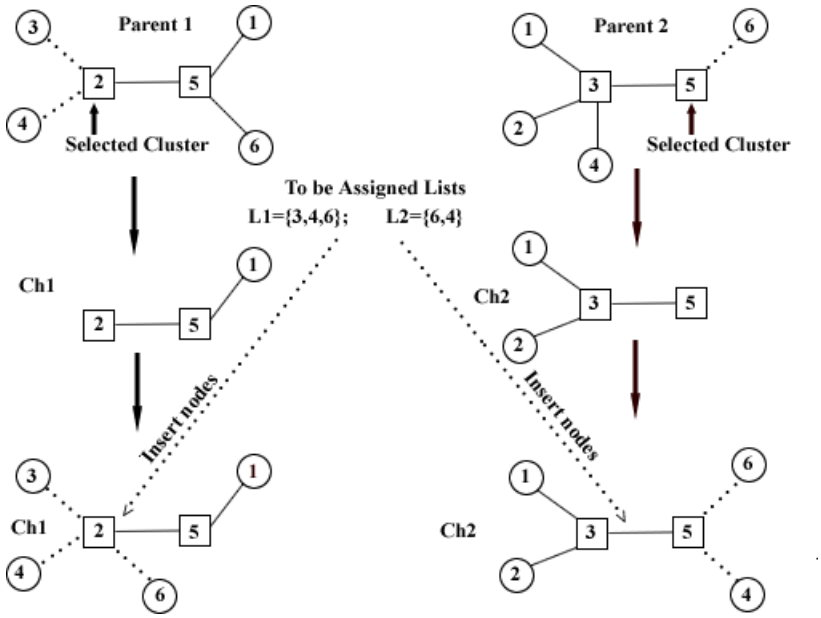

Figure 3.3: Illustration of the Random Exchange Spokes Crossover

3.3.3 Double Cluster Exchange Crossover (DCEC)

The DCEC was introduced in [25, 24]. Given two parents, DCEC randomly selects two clusters from one parent and inserts them into the offspring of the other parent, and then repeats the same operation for the second parent. Duplicate nodes resulting from the crossover operation are removed, and missing nodes are re-allocated to hubs that are closest to them. For CSAHLP, if the capacity of any hub is exceeded, its spokes are detached and re-allocated to other hubs until there is no excess flow. The final process checks for the existence of standalone hubs and demotes them by allocating those hubs to other hubs as spokes.

For illustration, consider two parents, P_1 and P_2, with $P_1 = \{\{\underline{4}, 1, 2\}, \{\underline{3}, 5, 9\}, \{\underline{8}, 6, 7, 10\}\}$ and $P_2 = \{\{\underline{6}, 1, 4\}, \{\underline{5}, 9, 10\}, \{\underline{8}, 7, 2, 3\}\}$ selected to undergo a DCEC. The underlined nodes in these representation are the hubs for the various clusters. The following steps illustrate how it works.

1. Randomly select two clusters from P_1, say, $P_1C_2 = \{\underline{3}, 5, 9\}$, and $P_1C_3 = \{\underline{8}, 6, 7, 10\}$, and repeat the same process for P_2, say $P_2C_1 = \{\underline{6}, 1, 4\}$, $P_2C_2 = \{\underline{5}, 9, 10\}$.

2. Delete the randomly selected clusters from their respective parents, and then copy the resulting parent's genetic information to their respective children, Ch_1 and Ch_2. This gives $Ch_1 = \{\{\underline{4}, 1, 2\}\}$ and $Ch_2 = \{\{\underline{8}, 7, 2, 3\}\}$.

3. Insert P_2C_1 and P_2C_2 into Ch_1. If a spoke in these clusters is a hub in Ch_1, the spoke is detached, otherwise it is deleted from Ch_1. If a hub from the clusters is also a hub in Ch_1, the two clusters are merged. The same process is repeated for Ch_2 using P_1. These produce $Ch_1 = \{\{\underline{4}, 1, 2\}, \{\underline{6}, 1\}, \{\underline{5}, 9, 10\}\}$, and $Ch_2 = \{\{\underline{8}, 7, 2, 3, 6, 10\}, \{\underline{3}, 5, 9\}\}$.

4. Missing nodes are allocated to hubs that are closest. In the case of CSAHLP, overflown hubs' spokes are detached and allocated to other non-overflown clusters until their capacity constraints are met. Standalone hubs (hubs without any spoke assigned to them) are demoted (assigned as spokes to other hubs). The created children are: $Ch_1 = \{\{\underline{4}, 2, 3, 7\}, \{\underline{6}, 1, 8\}, \{\underline{5}, 9, 10\}\}$, and $Ch_2 = \{\{\underline{8}, 7, 2, 3, 6, 10\}, \{\underline{3}, 5, 9, 1, 4\}\}$.

The above DCEC operation is illustrated in Figure 3.4.

3.3.4 Multi-Cluster Exchange Crossover (MCEC)

The MCEC, also introduced in the works of [25, 24], was employed in this study. Ideally, MCEC works the same as the DCEC except the number of clusters to be selected from each parent for the crossover varies in the range $1 \leq number\_of\_clusters \leq n-1$, where n is the number of clusters in the given parent. It first randomly determines the number of clusters to be selected, and then randomly selects this number from the parent. The selected cluster(s) are inserted into the opposite parent's child, and the same operation repeated for the opposite parent.

As an illustration, consider two parents, P_1 and P_2, with $P_1 = \{\{\underline{4}, 1, 2\}, \{\underline{3}, 5, 9\}, \{\underline{8}, 6, 7, 10\}\}$ and $P_2 = \{\{\underline{6}, 1, 4\}, \{\underline{5}, 9, 10\}, \{\underline{8}, 7, 2, 3\}\}$ selected to undergo a MCEC. The underlined nodes in these representation are the hubs for the various clusters. The following steps illustrate how it works.

1. Randomly select one or more clusters from each parent. In P_1, one cluster, $P_1C_2 = \{\underline{3}, 5, 9\}$, was selected, while two clusters, $P_2C_1 = \{\underline{6}, 1, 4\}$, $P_2C_2 = \{\underline{5}, 9, 10\}$, were selected from P_2.

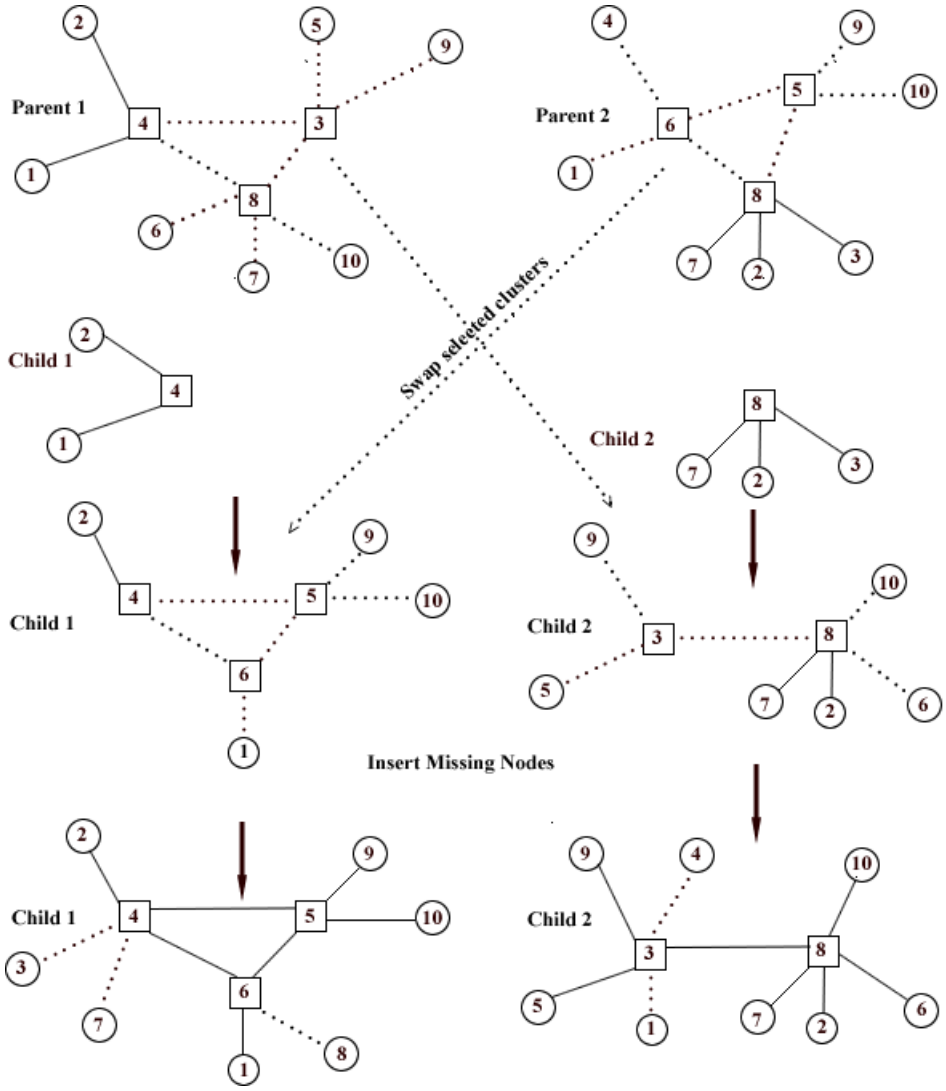

Figure 3.4: Illustration of the Double Cluster Exchange Crossover

2. Delete the randomly selected clusters from their respective parents, and then copy the resulting parent's genetic information to their respective children solutions, say, Ch_1, and Ch_2. This produces $Ch_1 = \{\{\underline{4}, 1, 2\}, \{\underline{8}, 6, 7, 10\}\}$, and $Ch_2 = \{\{\underline{8}, 7, 2, 3\}\}$.

3. Insert P_2C_1 and P_2C_2 into Ch_1. If a spoke in P_2C_1 and P_2C_2 is a hub in Ch_1, it is detached from its cluster before insertion, otherwise, it is detached from Ch_1. If a hub from either P_2C_1 or P_2C_2 is also a hub in Ch_1, the two clusters are merged. This process is repeated for Ch_2 using P_1C_2. These produce: $Ch_1 = \{\{\underline{4}, 2\}, \{\underline{6}, 1\}, \{\underline{5}, 9, 10\}, \{\underline{8}, 7\}\}$ and $Ch_2 = \{\{\underline{8}, 7, 2\}, \{\underline{3}, 5, 9\}\}$.

4. Missing nodes are allocated to hubs that are closest. In the case of CSAHLP,

overflown hubs' spokes are detached and allocated to other non-overflown clusters until their capacity constraints are met. Standalone hubs are also demoted. The created children are: $Ch_1 = \{\{\underline{4}, 2, 3\}, \{\underline{6}, 1\}, \{\underline{5}, 9, 10\}, \{\underline{8}, 7\}\}$, and $Ch_2 = \{\{\underline{8}, 7, 2, 6, 1\}, \{\underline{3}, 5, 9, 4, 10\}\}$.

The above processes of the MCEC are shown in Figure 3.5.

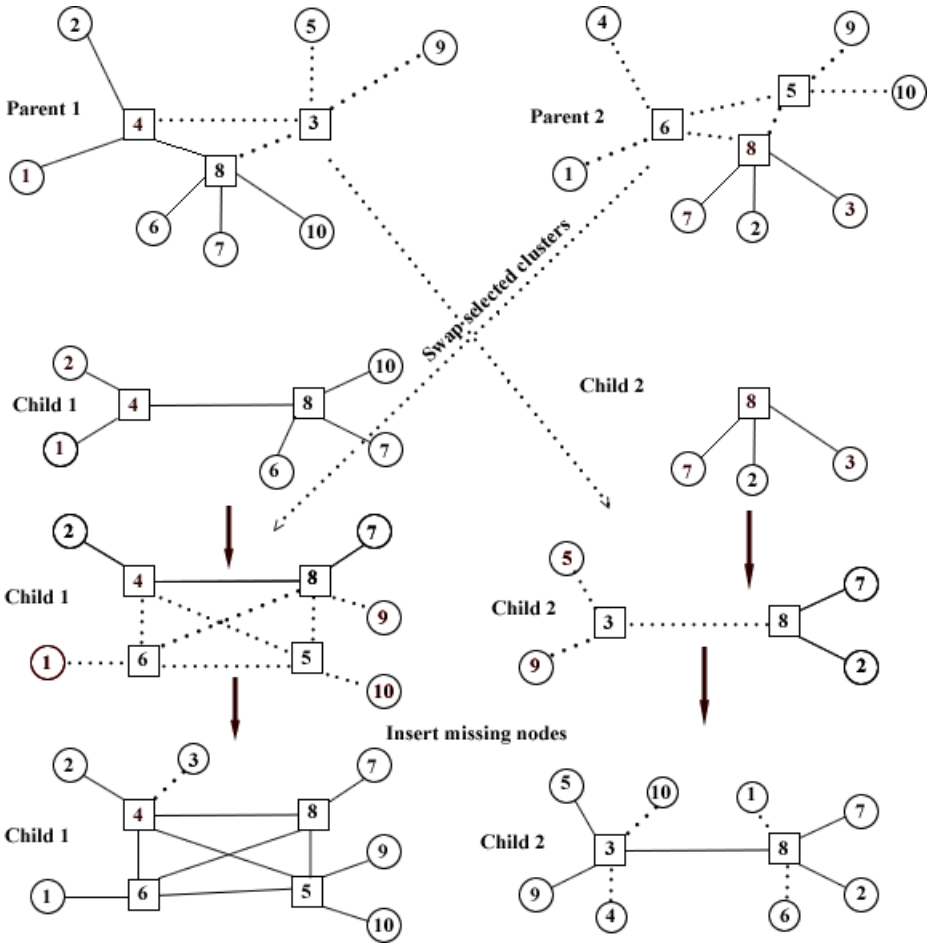

Figure 3.5: Illustration of the Multi-Cluster Exchange Crossover

3.4 Mutation

Three problem-specific mutation types were used probabilistically. These are *Shift Node* [1, 36], *Swap Hub* [1, 36], and *Replace Hub* [24, 25] mutations. Figure 3.6 illustrates these mutations. They are further explained below.

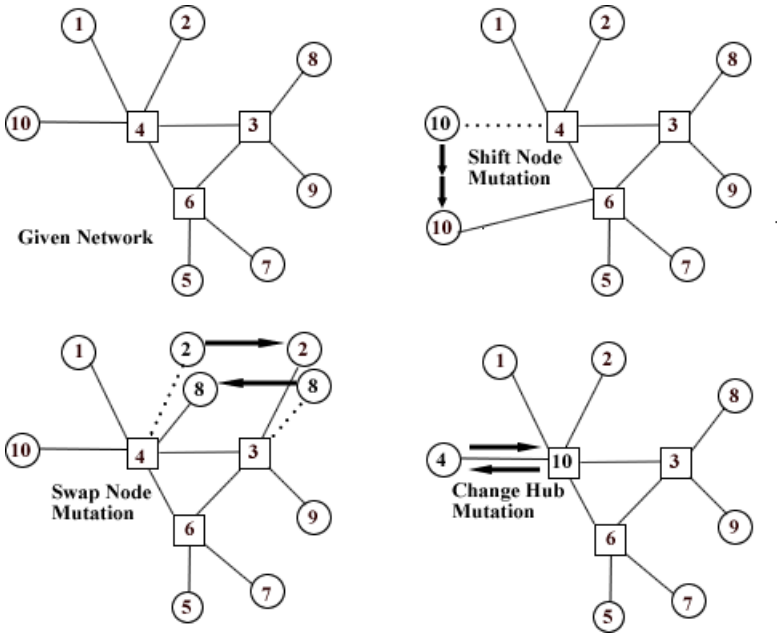

Figure 3.6: Illustration of the Three(3) Mutation Operators

3.4.1 Replace Hub Mutation

The Replace Hub mutation works by randomly selecting a cluster, and then demoting the hub of that cluster to a spoke, and promoting a randomly selected spoke in the cluster to become the hub of the cluster. For the CSAHLP, the capacity of the randomly chosen spoke must be greater than its flow in order to receive additional flow, else a spoke with higher capacity is chosen instead.

3.4.2 Swap Node Mutation

In a Swap Hub mutation, a random spoke each from two randomly selected clusters are exchanged. It is only performed if a given chromosome has more than one cluster.

3.4.3 Shift Node Mutation

The Shift Node mutation randomly selects a spoke from a randomly chosen cluster, and then allocates it to a different random cluster. This mutation is only performed if there is more than one cluster in chromosome, and the spoke's current cluster has more than one node.

3.5 Elitism

An elitism rate of 1% of the population was used. After the elite individuals were copied into the next generation, they were not deleted from the current population. This ensures that, during selection of chromosomes for reproduction, they also have the chance of being selected.

Chapter 4

Computational Results of the SOGA for the USAHLP

A detailed discussion and analysis of the experimental study of the SOGA for the USAHLP proposed in Chapter 3 is provided here.

4.1 Experimental set-up

The proposed SOGA was implemented in Microsoft Visual *C#* and all experiments performed on a 3GB Intel Core *i3* 64-bit Processor with speed 3GHz and 4GB RAM on a Windows 7 Home Premium environment. Four types of experiments were carried out with 50 runs for each problem instance as follows:

1. GA-BCRC - GA using BCRC
2. GA-RESC - GA using RESC
3. GA-DCEC - GA using DCEC
4. GA-MCEC - GA using MCEC

In each of the above experiments, any time mutation was to be applied, the *Shift Node* mutation was used at the rate of 0.2, *Swap Node* mutation at 0.6, while *replace-hub* mutation was used at the rate of 0.2. These rates were empirically determined.

4.1.1 GA Parameters

Table 4.1 shows the parameter settings that were empirically set for all the experimentations.

Table 4.1: GA Parameters for the SOGA

Parameter	Value
Population Size	300
Generational Span	1000
Number of Runs	50
Elitism Rate	1%
Crossover Rate	80%
Mutation Rate	20%
Tournament Size	3

4.2 Benchmark Data set

The two most commonly used data set for benchmarking the USAHLP are the AP [17] and CAB [26] data sets. These data sets are explained below.

4.2.1 CAB Data Set

CAB data set is based on the air traffic between 25 cities in the USA in the 1970s. It has problem instances of node sizes $n = 10$, 15, 20, and 25 with its traffic flow being symmetric ($W_{ij} = W_{ji}$). This data set does not have capacity restrictions on the hubs, and also does not include fixed costs on nodes. As a result, it is used only as test data for the USAHLP.

The *collection* (χ) and *distribution* (δ) unit costs for this data set are fixed at 1.0 whereas the *transfer* (α) cost varies at an interval of 0.2 for the range $0.2 \leq \alpha \leq 1$. Moreover, the cost of establishment of any node as a hub is fixed unlike the AP data set. The establishment cost vary at an interval of 50 for the range $100 \leq f \leq 250$.

4.2.2 AP Data Set

The AP data set was introduced by Ernst and Krishnamoorthy [17]. Each node in this data set denotes a postal district with their geographical coordinates and flow volumes in Australia. The AP data set has two versions: capacitated and uncapacitated. Since this chapter is concerned with the uncapacitated SAHLP, we discuss the uncapacitated version of the AP data set.

Two types of fixed costs exist for the uncapacitated AP data set: tight/high cost (T), and loose/low cost (L). Problems of type T are difficult to solve. Each problem with nodes size n has two instances: L, and T. In general, problems of the uncapacitated AP data set are denoted as nF where n is the problem instance's nodes size, and F is the fixed cost type. A problem instance denoted 10T means, a

tight-cost (T) problem instance with nodes size 10. The collection (χ), transfer (α), and distribution (δ) costs coefficients are respectively 3, 0.75 and 2 for this data set. Moreover, the inter-nodal flow of the uncapacitated AP is asymmetric ($W_{ij} \neq W_{ji}$) and any node can send data to itself ($W_{ii} \neq 0$).

The uncapacitated AP data set initially had problem instances of node sizes $n \leq 200$, but newer problems instances of up to 400 nodes have been introduced lately [31].

4.3 Experimental Results of the Uncapacitated AP Data Set

Experiments for problem instances of node sizes $n \leq 200$ were performed. In addition, the experiments were extended to the larger problem instances (300L, 300T, 400L, and 400T) introduced by Silva and Cunha [31].

In Table 4.2, the problem and its known best are represented in the first two columns. The next columns are the best solution out of 50 runs for each of the four crossover type implementation: GA-BCRC, GA-RESC, GA-DCEC, and GA-MCEC respectively. In the table, where the proposed GAs found the currently known best solution, it is indicated with a $\surd$, and the frequency of obtaining this known best in 50 runs is shown in the brackets next to the $\surd$. Where the proposed GA found better solutions than the currently known best, it is shown with a bold figure. In addition, the averages of the 50 runs are shown in Table A.1 of Appendix A.

From Table 4.2, it can be observed that all the crossovers' implementation obtained the currently known best for problems instances of node sizes $n \leq 200$. It can also be observed that GA-RESC, GA-DCEC and GA-MCEC consistently scored 50 out of 50 (the frequency of obtaining the known best in 50 runs) for problem instances 25T, 40T, 50T and 100T providing indications that, they had no difficulty finding the known best for problems of type "T". The GA-BCRC, though it did not score high frequencies for all the problem instances, proved to perform better than the rest when applied on the larger problem instances. Its results closely match that of Silva and Cunha [31], and found new best for problem instance 400T. The same observation made on the results of the other three crossovers show that they struggled in obtaining these currently known best.

Comparing the crossover types, GA-BCRC generally performed better than the rest and has proven to be robust when applied to problem instances of larger node

Table 4.2: Results of Uncapacitated AP Data Set & Comparison with Known Best

Problem	Known Best	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
10L	224,250.05	$\surd^{(23)}$	$\surd^{(3)}$	$\surd^{(2)}$	$\surd^{(3)}$
20L	234,690.95	$\surd^{(11)}$	$\surd^{(20)}$	$\surd^{(26)}$	$\surd^{(43)}$
25L	236,650.62	$\surd^{(4)}$	$\surd^{(22)}$	$\surd^{(30)}$	$\surd^{(42)}$
40L	240,986.23	$\surd^{(5)}$	$\surd^{(10)}$	$\surd^{(20)}$	$\surd^{(16)}$
50L	237,421.98	$\surd^{(8)}$	$\surd^{(29)}$	$\surd^{(45)}$	$\surd^{(45)}$
100L	238,016.28	$\surd^{(7)}$	$\surd^{(29)}$	$\surd^{(39)}$	$\surd^{(46)}$
200L	233,802.97	$\surd^{(8)}$	$\surd^{(18)}$	$\surd^{(29)}$	$\surd^{(40)}$
300L	264,837.88 [31]	$264,848.16^{(2)}$	$271,202.42^{(1)}$	$269,089.46^{(1)}$	$269,239.54^{(3)}$
400L	268,164.13 [31]	$268,631.09^{(1)}$	$275,621.46^{(1)}$	$272,715.14^{(4)}$	$275,540.36^{(2)}$
10T	263,399.94	$\surd^{(24)}$	$\surd^{(2)}$	$\surd^{(5)}$	$\surd^{(2)}$
20T	271,128.18	$\surd^{(8)}$	$\surd^{(2)}$	$\surd^{(6)}$	$\surd^{(3)}$
25T	295,667.84	$\surd^{(2)}$	$\surd^{(50)}$	$\surd^{(50)}$	$\surd^{(50)}$
40T	293,164.83	$\surd^{(11)}$	$\surd^{(50)}$	$\surd^{(50)}$	$\surd^{(50)}$
50T	300,420.98	$\surd^{(2)}$	$\surd^{(50)}$	$\surd^{(50)}$	$\surd^{(50)}$
100T	305,097.96	$\surd^{(20)}$	$\surd^{(50)}$	$\surd^{(50)}$	$\surd^{(50)}$
200T	272,188.11	$\surd^{(2)}$	$\surd^{(2)}$	$\surd^{(1)}$	$\surd^{(1)}$
300T	276,047.75 [31]	$279,178.29^{(4)}$	$290,854.54^{(2)}$	$284,993.86^{(1)}$	$291332.22^{(2)}$
400T	284,212.47 [31]	**284,124.88(2)**	$292,864.44^{(2)}$	$301,941.64^{(50)}$	$301,941.64^{(50)}$
Number of known best found		**15/18**	**14/18**	**14/18**	**14/18**

sizes since it got most of the currently known best, and established a new best solution for problem instance 400T.

A one-way Analysis of Variance (ANOVA) test to measure the statistical significance of differences in the means of the results of the crossover implementations was performed. The test used the results of the crossovers for the AP data set. The results of the ANOVA test shown in Table 4.3 reveal that there is statistically insignificant difference between the crossovers performances, hence providing indications that, no crossover outperformed the other.

Table 4.3: ANOVA test comparing the means of the GAs for the AP data set

	Degree of Freedom (df)	Sum Squared	Mean Squared	F value	P Value
X	3	4.512×10^9	1.504×10^9	1.804	0.155
Residuals	68	5.671×10^{10}	8.340×10^8		

4.3.1 Computational Time Performances of SOGA for the Uncapacitated AP Data set

A graph of the computational time of the GAs is shown in Figure 4.1. Since problem instances of type "T" are generally perceived to be the most difficult to solve, and

therefore requires higher computational time, the runtime of the GAs for this type of problems using node sizes $n = 20$, 40, 50, 100, and 200 were plotted. From the graph, it can be observed that BCRC steadily takes a longer time as the problem nodes size increases. This is followed by RESC. MCEC generally spends less computation time while DCEC is least from the initial stages. An observation from the chart shows that though generally the implementation of the BCRC proves robust in finding almost all the currently known-best and improving on others, it comes with high computational time. Therefore, if less computational time is required, then obviously, the best option will be to go for the MCEC or DCEC. However, if the goal is to have an approach that obtains the currently known-best for both small and larger problem instances, irrespective of the computational time, then BCRC should be chosen.

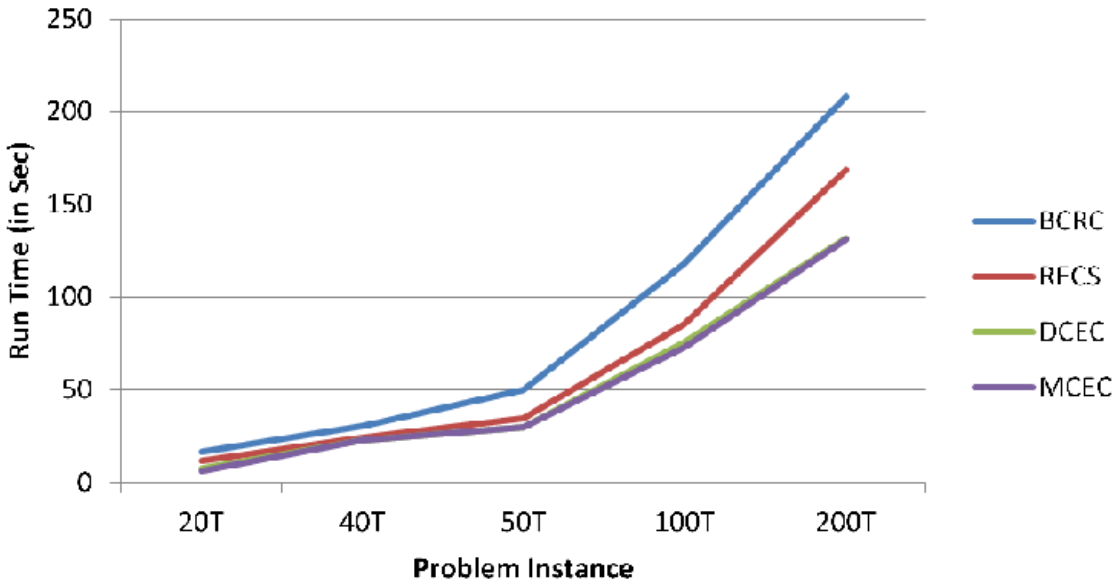

Figure 4.1: Computational Time of GAs for Problems of Type "T"

4.4 Experimental Results of the CAB Data Set

The results of applying the SOGA on the CAB data set are shown in Tables B.1, B.2, B.3, and B.4 of Appendix B. In these tables, the first two columns represent the α and the f values of the problem instance. The next columns represent the problem's currently known best (Optimal Cost), and the best solution obtained by each of the four crossover implementation types.

As it can be seen from these tables, out of the total of 80 problem instances, all the crossover operators, except GA-MCEC were effective in finding their currently

known best solutions. The GA-MCEC could not obtain the currently known best for problem instance $n = 15$ using f =100 and $\alpha = 0.2$ where it obtained 1110.09. These results confirm how competitive the crossover operators are when compared with the works of Silva and Cunha [31].

4.5 Conclusion

In conclusion, it can be observed from the above results that the implementation of the SOGA for the USAHLP has been effective and competitive in finding the currently known best solutions to problem instances of both the AP and CAB data sets.

Chapter 5

Computational Results of the SOGA for the CSAHLP

5.1 Experimental Set-up & GA Parameters

The computational results of the SOGA for the CSAHLP are presented here. The experimental set-up and GA parameter settings are the same as those used for the SOGA for the USAHLP in Chapter 4. A further description of the data set used (capacitated AP data set) is, however, provided here since only its uncapacitated version was explained in Chapter 4.

5.2 Benchmark Data set

As explained in section 4.2.1 of Chapter 4, the CAB data set does not factor in capacity restrictions on its hubs. Moreover, CAB does not include fixed costs on nodes. The AP data set, also explained in Section 4.2.2 of the same chapter, however, has capacity restrictions on the hubs as well as include fixed costs on nodes. As a result, the capacitated AP data set is the data set commonly used to benchmark CSAHLP, and therefore, was used in the experiments here.

5.2.1 The Capacitated AP Data Set

The capacitated AP data set has two types of capacities, in addition to the inclusion of the fixed costs on nodes as explained in Chapter 4. These capacity types are: tight (T), and loose (L). Problems of type "T" are difficult to solve while type "L" problems are relatively easy solving. A problem with node size n has four different

instances obtained through permuting the capacities and fixed costs types, thus, *LL, LT, TL, TT*. In general, problems of the capacitated AP data set are denoted as nFC where n is the number of nodes, F is the fixed cost type, and C represents the capacity type. For instance, a problem instance denoted $10LT$ represents a "Loose" fixed cost, "Tight" capacity type problem instance with node size 10. The collection (χ), transfer (α), and distribution (δ) costs coefficients are the same as that of the uncapacitated version. The capacitated AP data has problem instances of node sizes $n \leq 200$.

5.3 Results of the CSAHLP

SOGA was applied to various problem sizes of the capacitated AP data set and a comparative study done with best known solutions given in Contreras et al. [11]. The results are presented in Tables 5.1, 5.2, and 5.3 according to small, medium, and large data set. In these tables, the problem and its known best [11] are presented on the first two columns of the table. The same format of results is used for the other columns as those in Chapter 4. In addition, the averages of the 50 runs are shown in Tables A.2, A.3, and A.4 found in Appendix A.

5.3.1 Experimental Discussion

An analysis and comparison of the results obtained by each of the crossovers and that of the currently known best from [11] is made. Tables 5.1, 5.2, and 5.3 generally show that GA-DCEC and GA-MCEC were good at obtaining the currently known best for problem instances of the "LL" type with higher frequencies. From the results for the smaller problem instances (Table 5.1), GA-DCEC and GA-MCEC each scored 18 out of 20 whereas their counterparts GA-BCRC and GA-RESC each scored 20 out of 20 and 19 out of 20 respectively. Similar observations can be made for the other two tables.

Another observation is that all the four crossover operators were able to establish new best solutions (improved currently known best) for problem instance 25TT. GA-BCRC and GA-RESC, though generally scored lower frequencies, have proved to perform consistently well as the problem instance node sizes increase. In addition, most of the newly established best were in most cases found by these two crossover types implementation. Furthermore, GA-BCRC and GA-RESC generally performed better than the GA-DCEC and GA-MCEC since they got most of the currently

Table 5.1: Small Capacitated AP Data Set Results, and Comparison with Known Best [11]

Problem	Known Best [11]	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
10LL	224,250.05	$\surd^{(26)}$	$\surd^{(2)}$	$\surd^{(8)}$	$\surd^{(5)}$
10LT	250,992.26	$\surd^{(6)}$	$\surd^{(2)}$	$\surd^{(1)}$	$\surd^{(2)}$
10TL	263,399.94	$\surd^{(37)}$	$\surd^{(4)}$	$\surd^{(8)}$	$\surd^{(7)}$
10TT	263,399.94	$\surd^{(37)}$	$\surd^{(4)}$	$\surd^{(8)}$	$\surd^{(7)}$
20LL	234,690.94	$\surd^{(12)}$	$\surd^{(46)}$	$\surd^{(50)}$	$\surd^{(36)}$
20LT	253,517.40	$\surd^{(6)}$	$\surd^{(27)}$	$\surd^{(44)}$	$\surd^{(50)}$
20TL	271,128.18	$\surd^{(7)}$	$\surd^{(2)}$	$\surd^{(3)}$	$\surd^{(5)}$
20TT	296,035.40	$\surd^{(4)}$	$\surd^{(29)}$	$\surd^{(14)}$	$\surd^{(25)}$
25LL	238,977.95	$\surd^{(3)}$	$\surd^{(27)}$	$\surd^{(25)}$	$\surd^{(33)}$
25LT	276,372.50	$\surd^{(2)}$	$\surd^{(5)}$	$\surd^{(11)}$	$\surd^{(12)}$
25TL	310,317.64	$\surd^{(4)}$	$\surd^{(3)}$	$\surd^{(21)}$	$\surd^{(31)}$
25TT	348,369.15	**346,582.01**	**346,582.01**	**348,280.00**	**346,582.01**
40LL	241,955.71	$\surd^{(3)}$	$\surd^{(35)}$	$\surd^{(18)}$	$\surd^{(23)}$
40LT	272,218.32	$\surd^{(5)}$	$\surd^{(6)}$	$\surd^{(4)}$	$\surd^{(30)}$
40TL	298,919.01	$\surd^{(4)}$	$\surd^{(19)}$	$\surd^{(23)}$	$\surd^{(27)}$
40TT	354,874.10	$\surd^{(2)}$	$\surd^{(2)}$	356,509.86	356,453.79
50LL	238,520.59	$\surd^{(7)}$	$\surd^{(13)}$	$\surd^{(50)}$	$\surd^{(50)}$
50LT	272,897.49	$\surd^{(8)}$	$\surd^{(1)}$	$\surd^{(2)}$	$\surd^{(5)}$
50TL	319,015.77	$\surd^{(4)}$	$\surd^{(2)}$	$\surd^{(14)}$	$\surd^{(7)}$
50TT	417,440.99	$\surd^{(1)}$	418,550.34	422,064.22	419,510.01
No. of known best found		**20/20**	**19/20**	**18/20**	**18/20**

known best, and in most cases, found new best solutions for some of the problem instances. However, a one-way ANOVA test performed on the means of 50 runs for each problem instance of the capacitated AP data set obtained using the four crossovers shows otherwise. The test results are shown in Table 5.4. From this table, the ANOVA p-value shows that there is statistically insignificant difference among the performances of the four crossovers.

5.3.2 Computational Time Performances of the SOGA for CSAHLP

A graph showing the computational time performances of the GAs is shown in Figure 5.1. The graph is a plot of the runtime for problem instance "TT" for node sizes $n =$

Table 5.2: Medium Capacitated AP Data Set Results & Comparison with Known Best

Problem	Known Best [11]	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
60LL	225,917.21	$\surd^{(4)}$	$\surd^{(50)}$	$\surd^{(49)}$	$\surd^{(48)}$
60LT	253,761.98	257,547.85	254,280.26	257,275.30	255,906.22
60TL	252,496.66	$\surd^{(10)}$	$\surd^{(45)}$	$\surd^{(44)}$	$\surd^{(32)}$
60TT	351,274.72	$\surd^{(1)}$	**351,203.17**	351,548.54	351,594.70
70LL	236,817.35	$\surd^{(1)}$	$\surd^{(24)}$	$\surd^{(16)}$	$\surd^{(13)}$
70LT	257,454.36	**257,207.70**	**256,939.56**	$\surd^{(1)}$	$\surd^{(1)}$
70TL	271,283.82	$\surd^{(6)}$	$\surd^{(28)}$	$\surd^{(33)}$	$\surd^{(31)}$
70TT	387,380.20	$\surd^{(2)}$	387,836.00	388,362.07	387,820.07
75LL	238,024.22	$\surd^{(4)}$	$\surd^{(37)}$	$\surd^{(35)}$	$\surd^{(33)}$
75LT	256,188.12	$\surd^{(1)}$	$\surd^{(2)}$	$\surd^{(7)}$	$\surd^{(1)}$
75TL	303,363.55	$\surd^{(3)}$	$\surd^{(6)}$	$\surd^{(26)}$	$\surd^{(28)}$
75TT	347,189.81	347,563.67	347,907.08	349,421.19	348,555.35
90LL	224,195.72	$\surd^{(3)}$	$\surd^{(35)}$	$\surd^{(50)}$	$\surd^{(50)}$
90LT	246,026.24	$\surd^{(1)}$	246,850.51	248,388.02	246,613.76
90TL	281,561.56	281,888.74	282,304.23	298,839.80	298,839.80
90TT	337,008.93	337,540.73	338,077.13	343,662.44	341,207.01
No. of known best found		**12/16**	**10/16**	**9/16**	**9/16**

20, 40, 50, 70, 100, 125, 150, 175, and 200. From the graph, it can be observed that BCRC steadily takes longer in computation time as the problem nodes size increases. This is followed by RESC. MCEC and DCEC initially appear to be spending the same and less computation time but as the node size increases, DCEC fluctuates and steadily gets back to be the same as MCEC.

The graph also shows that though generally the implementation of the BCRC proved robust in finding almost all the currently known best for most problems, and establishing new best solutions for other problem instances, it always does so at high computational time. Therefore, if less computation time is required, then the best option will be to go for the MCEC or DCEC. However, if the goal is to have an approach that obtains the currently known-best for all problem types (small and large) as well as establish new solutions for some problems irrespective of the computational time, then BCRC stands out. An alternative option to the BCRC is the RESC since its performance proved robust after that of the BCRC.

Table 5.3: Large Capacitated AP Data Set Results & Comparison with Known Best [11]

Problem	Known Best [11]	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
100LL	246,713.97	$\surd^{(2)}$	$\surd^{(2)}$	$\surd^{(5)}$	$\surd^{(4)}$
100LT	256,155.33	256,639.38	256,237.99	256,553.74	256,355.96
100TL	362,950.09	$\surd^{(4)}$	$\surd^{(1)}$	$\surd^{(1)}$	$\surd^{(2)}$
100TT	474,680.32	$\surd^{(1)}$	$\surd^{(1)}$	475,344.62	475,344.62
125LL	239,920.75	**239,889.33**	**239,889.33**	**239,889.33**	$\surd^{(2)}$
125LT	251,259.16	$\surd^{(2)}$	$\surd^{(2)}$	257,288.70	251,285.88
125TL	246,486.69	$\surd^{(4)}$	$\surd^{(50)}$	$\surd^{(38)}$	$\surd^{(48)}$
125TT	291,807.35	$\surd^{(2)}$	293,265.26	309,697.22	297,523.88
150LL	234,765.44	$\surd^{(5)}$	$\surd^{(44)}$	$\surd^{(36)}$	$\surd^{(39)}$
150LT	250,186.53	$\surd^{(1)}$	$\surd^{(2)}$	257,723.68	251,354.59
150TL	262,822.24	**262,583.31**	**262,716.52**	263,502.69	**262,583.31**
150TT	323,992.37	**323,777.37**	327,052.32	324,091.43	**323,777.37**
175LL	227,997.58	$\surd^{(2)}$	$\surd^{(16)}$	$\surd^{(3)}$	$\surd^{(9)}$
175LT	251,540.80	261,459.73	253,900.23	259,843.31	253,124.42
175TL	244,860.41	$\surd^{(5)}$	$\surd^{(7)}$	$\surd^{(3)}$	$\surd^{(22)}$
175TT	308,310.13	317,160.29	317,885.27	320,716.80	317,894.61
200LL	231,069.50	241,992.97	241,992.97	241,992.97	241,992.97
200LT	268,820.57	**268,682.61**	269,065.28	**268,682.61**	**268,682.61**
200TL	273,443.81	$\surd^{(7)}$	$\surd^{(2)}$	276,369.59	279,915.93
200TT	290,841.84	297,099.13	294,694.45	315,371.41	294,838.03
No. of known best found		**15/20**	**12/20**	**8/20**	**10/20**

Table 5.4: ANOVA test comparing the means of the GAs for AP data set

	Degree of Freedom (df)	Sum Squared	Mean Squared	F value	P Value
X	3	2.168×10^{10}	7.225×10^{9}	2.198	0.0892
Residuals	220	7.233×10^{11}	3.288×10^{9}		

5.4 Conclusion

In conclusion, a summary of the performances of the GAs is provided in Tables 5.5, 5.6 and 5.7. It is clear from these tables that SOGA has been effective and therefore, most suited for applying to the CSAHLP. Through SOGA's approach, new solutions have been established for problem instances 25TT, 60TT, 70LT, 125LL, 150TL, 150TT, and 200LT.

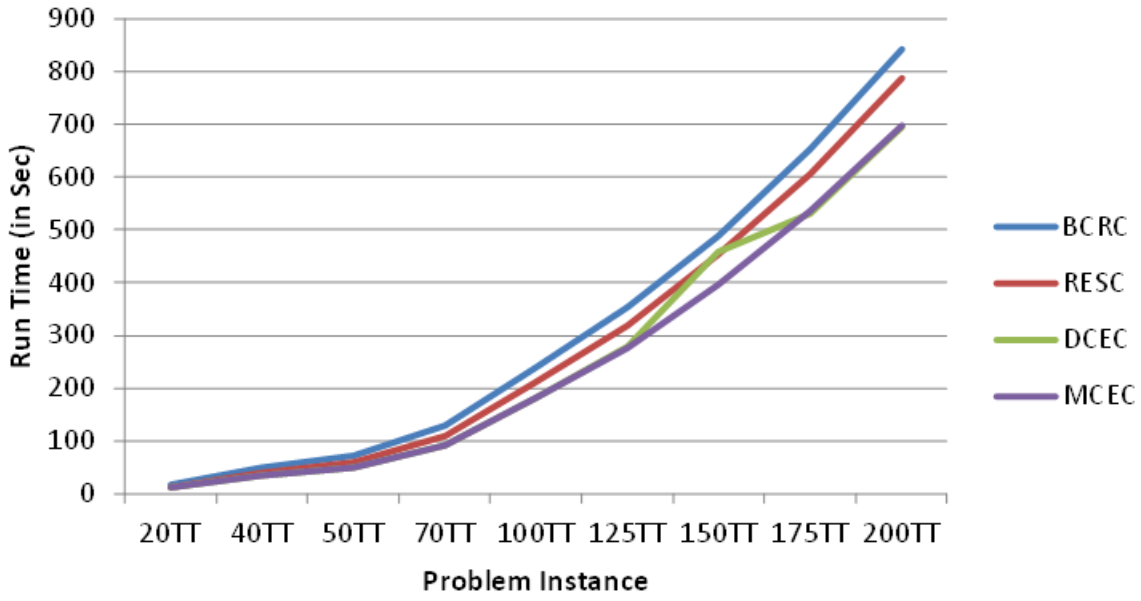

Figure 5.1: Plot of computational Time of GAs for Problems of Type "TT"

Table 5.5: Summary of SOGA Performances for Small Problem Instances

	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
Currently Known Best	19	18	17	17
New Best	1	1	1	1
Inferior	0	1	2	2
Total Score	**20**	**19**	**18**	**18**
Number of Problem Instances	20	20	20	20

Table 5.6: Summary of SOGA Performances for Medium Problem Instances

	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
Currently Known Best	11	8	9	9
New Best	1	2	0	0
Inferior	4	6	7	7
Total Score	**12**	**10**	**9**	**9**
Number of Problem Instances	16	16	16	16

Table 5.7: Summary of SOGA Performances for Large Problem Instances

	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
Currently Known Best	11	10	6	7
New Best	4	2	2	3
Inferior	5	8	12	10
Total Score	**15**	**12**	**8**	**10**
Number of Problem Instances	20	20	20	20

Chapter 6

Multi-Objective Genetic Algorithm for the CSAHLP

This chapter discusses the design and implementation of the MOGA for the CSAHLP.

6.1 Multi-Objective CSAHLP

Most previous works on the CSAHLP have focused on minimizing the total cost of network transportation while ensuring that the capacities of the various hubs are not exceeded. That is, their approaches have only one objective: minimising the total network transportation cost. This, however, limits the decision maker (DM) to just two options: the acceptance or rejection of the proposed solutions. As a result, an approach that provides more options to choose from, is most appropriate.

In overcoming these limitations, Costa et al. [12] introduced the idea of removing the capacity constraints on the CSAHLP, and introducing a second objective measure, called the time to process flow received at the various hubs, in addition to the first objective function: the network transportation cost. Their work was motivated by the fact that the non-dominated solutions (n.d.s) provided by the multi-objective approach would serve as good support decision tool for managerial decisions as compared to the single objective approach. Their approach used an integer linear programming strategy.

In this thesis, a MOGA approach, motivated by [12], is proposed. The MOGA approach was also motivated by the fact that many times situations arise where the cost of re-routing any excess flow to a different hub is more costly than making adjustment at the current over-flown hub to accommodate the excess flow. This is practical in many real life situations. For instance, in a health care delivery system, if

on admission, a patient's survival rate depends on how early he/she is attended to by a medical team, the time to re-direct him/her to another health centre just because the current place has exceeded its capacity can cost the patient's life. This situation could be minimised if logistic adjustments are made by extending the service time. In such a case, the removal/violation of the capacity constraint is recommended and inevitable.

6.1.1 Multi-Objective Formulation

From the works of Costa et al. [12], the problem formulation of the multi-objective CSAHLP are shown in Equations 6.1 and 6.2 below. In these formulations, T_k is the time hub k takes to process one unit of flow; P_k is the fixed time to initiate service at hub k; while the rest of the variables in the formulations retain their meanings from Section 2.2.1 of Chapter 2. Moreover, Constraints (2.5) and (2.6) from the formulations in Chapter 2 have been removed, and a second objective function which measures the total time to process flow gathered at the various hubs is introduced.

$$\min \sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n}\sum_{j=1}^{n} W_{ij}(\chi \times d_{ik} + \alpha \times d_{kl} + \delta \times d_{lj})X_{ijkl} + \sum_{k=1}^{n} F_k Z_{kk} \tag{6.1}$$

$$\min \sum_{i=1}^{n}\sum_{k=1}^{n} O_i T_k Z_{ik} + \sum_{k=1}^{n} P_k Z_{kk} \tag{6.2}$$

Subject to:

$$\sum_{k=1}^{n} Z_{ik} = 1,\ \forall i, k \in n, \tag{6.3}$$

$$Z_{ik} \leq Z_{kk},\ \forall i, k \in n, \tag{6.4}$$

6.2 MOGA for the CSAHLP

The MOGA algorithm, shown in Algorithm 6, differs from SOGA by only the fitness evaluation strategy, where the multi-objective fitness evaluation strategies described in Section 2.5 of Chapter 2 are employed. During chromosome evaluation, one of these evaluation strategies is used to evaluate the overall fitness of the chromosome based on the computed transportation cost and processing time. Furthermore, only

the BCRC crossover is used here. Apart from when weighted sum fitness evaluation strategy is used, the output of MOGA is a list of n.d.s. These n.d.s are the ones being analysed and provided to the DM. In the case of the weighted sum evaluation approach, only one solution is provided by the MOGA.

Algorithm 6: MOGA for the CSAHLP

```
begin
    Read problem instance data set
    Set GA parameters
    Randomly create initial population
    while termination criteria is not met do
        foreach chromosome in population do
            Compute its transportation cost, C(x), and processing time, T(x)
            Evaluate its fitness using Pareto, Summed Ranks, or Weighted Sum
        end
        Add elite chromosomes to new population using elitism rate
        while the size of new population < population size do
            Select chromosomes for reproduction using tournament selection
            scheme
            Apply crossover using crossover rate
            Apply mutation using mutation rate
            Add offspring to new population
        end
        Replace old population with new population
    end
    Return the list of n.d.s
end
```

6.3 Objective Functions

Two objective dimensions were used in measuring the fitnesses of the chromosomes. These objective measures are:

1. *Network transportation cost:* shown in Equation 6.5, this measures the overall network transportation cost of flow. In the equation, the cost of the internodal flow is the first term, while the hub establishment cost is represented by the second term. Algorithm 4 in Chapter 3 shows how to compute it.

$$C(x) = \sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n}\sum_{j=1}^{n} W_{ij}(\chi \times d_{ik} + \alpha \times d_{kl} + \delta \times d_{lj})X_{ijkl} + \sum_{k=1}^{n} F_k Z_{kk} \tag{6.5}$$

2. *Processing Time:* this measures the time to process the flow gathered at the hubs of the hub-and-spoke network. In computing it, some assumptions were made since the AP data set does not include processing times. As a result, the assumptions and formulation used in [12] to compute these values were employed. In [12], a given hub's capacity is expressed as the amount of flow it can process in a given day consisting of eight (8) hours. In expressing time in seconds, a day consists of 28800 seconds. In addition, there is a constant time to initiate service at any given hub, and these components together constitute the processing time of any given hub in the hub-and-spoke network. In Equation 6.6, the computation of this measure for a given hub k is shown. In this equation, the first term represents the time hub k takes to process flow gathered at it, and the second term represents the fixed time to initiate service at this hub. This formulation is from [12].

$$T(x) = \sum_{i=1}^{n}\sum_{k=1}^{n} O_i T_k Z_{ik} + \sum_{k=1}^{n} P_k Z_{kk} \tag{6.6}$$

In computing this value in the MOGA, Algorithm 7 is employed. In this algorithm, T_p is the total processing time; h_s, h_f, and h_c are the service time, total flow, and capacity of hub h respectively; while *hubList* is the list of hubs in the given network.

6.4 Fitness Evaluation Strategies

The fitness evaluation strategies employed for the MOGA are explained next.

Algorithm 7: Processing Time Algorithm

begin
  Input: Chromosome
  Output: Total Processing Time, T_p
  $T_p \leftarrow 0$
  foreach *h in hubList* **do**
    $T_p \leftarrow T_p + h_s(h) + 28800/h_c(h) * h_f(h)$
  end
  return T_p
end

6.4.1 Weighted Sum

The evaluation strategy of the weighted sum works by adding the given objective measures to obtain a single objective function by incorporating weights that provide a balance measure among the given objective measures [27, 28]. In this work, the two objectives are inversely related. Choosing the appropriate weight parameters for these measures is influenced by this relationship. By incorporating these weight parameters into the objective function, Equation 6.7 was obtained. In the formulation, $C(x)$ is the network transportation cost; $T(x)$ is the time to process flow; α and β are respectively the weight parameters for $C(x)$, and $T(x)$.

$$F(x) = \alpha \cdot C(x) + \beta \cdot (1/T(x)) \tag{6.7}$$

In this work, α and β were empirically established to be 1 and 1000000 respectively, and these values were used in all the experiments with this evaluation strategy.

6.4.2 Pareto Ranking

The Pareto Ranking strategy was employed as one of the fitness evaluation strategies for the MOGA. After computing the transportation costs, and processing times of the chromosomes, this strategy was used to rank the chromosomes. After the ranking process, the raw fitnesses of the chromosomes were replaced by these ranks and the selection of chromosomes for reproduction then performed using the ranks as fitnesses.

6.4.3 Sum of Ranks

The normalized sum of ranks, explained in Chapter 2, was also employed in MOGA. It was applied after the raw fitnesses of the chromosomes were computed. The re-

sulting ranks after this process replaced the raw fitnesses of the chromosomes, and tournament selection done using the ranks as fitness values.

Chapter 7

Computational Results of the MOGA for the CSAHLP

The computational results of MOGA for the CSAHLP are provided here.

7.1 Experimental Set-up

Three sets of experiments were carried out, namely

1. Pareto Ranking - MOGA using the Pareto Ranking fitness evaluation strategy
2. Sum of Ranks - MOGA using the Sum of Ranks fitness evaluation strategy, and
3. Weighted Sum - MOGA using Weighted Sum fitness evaluation strategy.

The experimental test data set and GA parameters are explained in the next sections.

7.2 Benchmark Data set

Since MOGA was proposed for the CSAHLP, the capacitated AP data set described in sub-section 5.2.1 of Chapter 5 was used. An empirical study on problem instances of nodes sizes n =10, 20, 25, and 40 giving a total of 16 problems (each node size has four different instances) was made, and a comparative analysis of the computed results made with that of the works of Costa et al. [12].

7.2.1 Categorization of Problem Instances

The works of Costa et al. [12] classified the 16 problem instances into three groups:

- Group I: Problem instances that have their optimal solutions of the SOGA to belong to the list of n.d.s from the MOGA and they include 10LT, 20LT, 25LL, 25TL, and 40LL. With this group, the n.d.s from the MOGA depict that one of the two objectives can be improved at the expense of the other. They generally have lower cost, and lower service times. The n.d.s provide options to the DM on how much flow could be exceeded at a processing center to improve the cost or time

- Group II: Problem instances 10LL, 10TL, 10TT, 20LL, and 20TL exhibit features of this group of problems. Their capacity constraints are not restrictive and therefore can be removed without any violations. They also tend to yield lower service times at higher cost. An advantage of classifying problems in this group is that, it gives the DM options of service quality improvement verses cost minimization. If the former is of prime importance, then he/she does that at the expense of the latter, and vice-versa.

- Group III: Here, the optimal solution of the SOGA match the dominated solution of the MOGA, and there is excessive flow volumes on the hubs. Problem instances 20TT, 25LT, 25TT, 40LT, 40TL, and 40TT fall under this group. The MOGA results present results that allow for improvement of both objective measures at the same time with excess flow.

7.3 GA Parameters

The parameter values shown in Table 7.1 below were empirically determined and used for all experimentations.

7.4 Computational Results

To demonstrate the effectiveness of the MOGA approach as an alternative to the SOGA, and also to compare the results with [12], the trade-offs of the transportation costs and the processing times were computed. The trade-offs, expressed in percentages, were computed as the difference between the objective values obtained in the

Table 7.1: GA Parameters for MOGA

Parameter	Value
Population Size	300
Generational Span	200
Number of Runs	50
Elitism Rate	1%
Crossover Rate	80%
Mutation Rate	20%
Tournament Size	3

MOGA, and that of the SOGA. They ideally provide an overview of how one objective criteria could be exploited at the expense of the other.

Equations 7.1 and 7.2 respectively show how the trade-offs of the transportation cost and processing time were computed. In these equations, f^1_{MOGA} and f^2_{MOGA} respectively represent the values of the transportation cost, and processing times obtained from the MOGA; f^1_{SOGA} and f^2_{SOGA} are the values of the transportation cost and processing times respectively obtained from the SOGA; while $\Delta f1(\%)$ and $\Delta f2(\%)$ represent the trade-offs of the transportation cost and processing times respectively.

After computing these trade-offs, an analysis is made for the three sets of experiments.

$$\Delta f1(\%) = \frac{f^1_{MOGA} - f^1_{SOGA}}{f^1_{SOGA}} \times 100\% \tag{7.1}$$

$$\Delta f2(\%) = \frac{f^2_{MOGA} - f^2_{SOGA}}{f^2_{SOGA}} \times 100\% \tag{7.2}$$

In the following sub-sections, the results for the three experiments are provided. The results are grouped into the three groups of problem instances as outlined in Section 7.2.1.

7.4.1 Pareto Ranking

The output of the MOGA using this strategy is a list of non-dominated solutions (n.d.s). A total of 50 runs were performed for each of the 16 problem instances and the best solutions set from these runs are those presented and discussed here. From these solutions, four were chosen for analysing each problem instance and to allow for an easier comparison with [12]. Details on the number of n.d.s provided by the

Pareto Ranking strategy and other breakdowns are shown in Tables 7.2, 7.3 and 7.4.

Tables 7.5, 7.6, and 7.7 show the computed results of the Pareto Ranking fitness evaluation strategy, and the results from [12]. In these tables, each problem instance is presented with four n.d.s within which the decision maker could choose from. Moreover, "Bi-criteria" represents the results from [12], and "Pareto"represents the results of the Pareto Ranking.

The n.d.s carefully selected can provide alternative options to the decision maker depending on where managerial priorities lie, unlike the single solution provided by the SOGA for a given problem instance. Some n.d.s allow for improvement of both objective measures; others improve one objective at the expense of the other. From Tables 7.5, 7.6 and 7.7, these observations can be made. The results of this fitness evaluation scheme clearly depict how competitive the MOGA approach is compared with that of [12]. The results also provide us with more convincing reasons why a MOGA approach using the Pareto Ranking strategy is a better alternative to the single solution of a SOGA for a given problem instance since the DM has at least four solutions to choose from.

Moreover, the values shown in Tables 7.5, 7.6 and 7.7 provide enough information on how cost can be improved at the expense of service time or the vice-versa, and at what excess of flow, that the given network's hub(s) would have to accommodate in order to achieve it. In particular, the results in Table 7.6 deserve some attention. A careful look at the *"Number of Hubs with flow Excess"* column reveal that there is no excess flow from the solutions provided by the MOGA. This gives indications that the capacity constraints placed on the hubs of these set of problems (categorized as Group II) can be removed. However, as it can be seen in this table, doing so (removing the capacity constraints) will only come at an increased cost.

Table 7.2: Group I: Comparing Characteristics of problems using Pareto Ranks' Solutions with [12]

Problem	Number of n.d.s		Number of n.d.s that improve f_1 and f_2		Number of n.d.s that improve only f_1		Number of n.d.s that improve only f_2	
	Bi-criteria	Pareto Ranks	Bi-criteria	Pareto Ranks	Bi-criteria	Pareto Ranks	Bi-criteria	Pareto Ranks
10LT	22	8	11	2	10	4	0	2
20LT	17	30	10	15	15	9	14	6
25LL	31	24	5	11	1	4	6	9
25TL	29	52	1	10	19	23	22	7
40LL	42	10	3	1	0	8	15	1

A plot showing how the individuals in the population are stratified as a result of the Pareto Ranking procedure is shown in Figure 7.1.

Table 7.3: Group II: Comparing Characteristics of problems using Pareto Ranks' Solutions with [12]

Problem	Total Number of n.d.s		Number of n.d.s with excess Flow	
	Bi-criteria	Pareto Ranks	Bi-criteria	Pareto Ranks
10LL	13	10	0	0
10TL	8	11	0	0
10TT	12	9	7	0
20LL	23	15	0	0
20TL	19	10	0	0

Table 7.4: Group III: Comparing Characteristics of problems using Pareto Ranks' Solutions with [12]

Problem	Number of n.d.s		Number of n.d.s that improve f_1 and f_2		Number of n.d.s that improve only f_1		Number of n.d.s that improve only f_2	
	Bi-criteria	Pareto Ranks	Bi-criteria	Pareto Ranks	Bi-criteria	Pareto Ranks	Bi-criteria	Pareto Ranks
20TT	6	8	1	2	5	4	0	2
25LT	38	30	9	15	15	9	14	6
25TT	25	24	18	11	1	4	6	9
40LT	52	52	11	10	19	23	22	7
40TL	16	10	1	1	0	8	15	1
40TT	26	20	9	9	8	5	9	6

Table 7.5: Group I: Pareto Ranking's Results & Comparison with Costa et al. [12]

Problem	$\Delta f1$ (%)		$\Delta f2$ (%)		Maximum Excess of flow (%)		Number of Hubs with flow Excess		Total Number of Hubs in Solution	
	Bi-criteria	Pareto	Bi-criteria	Pareto	Bi-criteria	Pareto	Bi-criteria	Pareto	Bi-criteria	Pareto
10LT	-10.65	10.82	520.72	-32.55	752.28	0.00	2	0	3	2
	-1.79	-1.79	64.83	64.83	47.94	47.94	1	1	3	3
	1.30	6.50	-6.24	-31.20	0.00	0.00	0	0	3	2
	2.90	3.22	-7.89	3.47	0.00	0.00	0	0	3	2
20LT	-2.72	-2.72	0.85	0.85	15.04	15.04	1	1	2	2
	-1.31	-6.55	0.29	237.35	4.44	23.94	1	1	2	3
	0.47	-1.29	-0.56	0.57	2.95	0.00	1	0	2	2
	8.53	8.83	-6.63	-6.63	82.21	82.21	1	1	1	3
25LL	-0.97	18.75	0.94	-34.17	15.04	27.15	1	1	2	3
	-0.26	17.80	0.19	-28.10	0.76	0.43	1	1	2	2
	2.60	2.60	-18.87	-18.87	0.00	0.00	0	0	2	2
	2.84	4.34	-19.12	-17.09	0.00	0.00	0	0	2	2
25TL	-4.72	-4.72	97.92	97.92	137.34	137.34	1	1	1	2
	12.30	18.94	-9.25	-1.14	0.00	0.00	0	0	2	2
	15.66	14.38	-10.94	2.25	0.00	0.00	0	0	2	3
	18.76	19.39	-16.20	-1.47	0.00	0.00	0	0	1	3
40LL	-0.40	5.75	296.22	-19.98	534.82	317.71	1	1	2	1
	-0.29	4.46	287.75	-22.97	516.86	558.28	1	1	2	1
	0.07	0.07	-21.33	-21.33	0.00	0.00	0	0	2	2
	0.23	2.57	-21.96	-5.10	0.00	0.00	0	0	2	4

Table 7.6: Group II: Pareto Ranking's Results & Comparison with Costa et al. [12]

Problem	$\Delta f1$ (%)		$\Delta f2$ (%)		Maximum Excess of flow (%)		Number of Hubs with flow Excess		Total Number of Hubs in Solution	
	Bi-criteria	Pareto	Bi-criteria	Pareto	Bi-criteria	Pareto	Bi-criteria	Pareto	Bi-criteria	Pareto
10LL	0.77	6.85	-8.19	-16.68	0.00	0.00	0	0	3	3
	1.58	1.58	-10.31	-10.31	0.00	0.00	0	0	3	2
	2.57	2.57	-18.36	-18.36	0.00	0.00	0	0	2	2
	3.76	3.76	-7.89	-7.89	0.00	0.00	0	0	2	2
10TL	0.43	0.43	-6.03	-6.03	0.00	0.00	0	0	2	2
	2.13	2.13	-7.29	-7.29	0.00	0.00	0	0	2	2
	4.15	12.50	-8.81	-21.60	0.00	0.00	0	0	2	2
	9.55	8.59	-16.81	-8.46	0.00	0.00	0	0	2	2
10TT	1.53	1.53	-1.76	-1.76	0.00	0.00	0	0	3	3
	4.04	9.31	-6.32	-3.17	0.00	0.00	0	0	2	3
	5.23	5.23	-6.50	-7.10	0.00	0.00	0	0	2	2
	7.07	7.07	-6.51	-6.51	0.00	0.00	0	0	2	2
20LL	4.26	4.26	-7.02	-7.02	0.00	0.00	0	0	2	2
	6.53	6.53	-11.13	-11.13	0.00	0.00	0	0	2	2
	6.98	6.98	-11.50	-11.50	0.00	0.00	0	0	2	2
20TL	0.39	0.39	-0.42	-0.42	0.00	0.00	0	0	2	2
	1.58	2.81	-0.84	-0.74	0.00	0.00	0	0	2	2
	2.60	2.60	-5.65	-5.65	0.00	0.00	0	0	1	1
	15.57	16.39	-10.82	0.00	0.00	0.00	0	0	2	3

Table 7.7: Group III: Pareto Ranking's Results & Comparison with Costa et al. [12]

Problem	$\Delta f1$ (%)		$\Delta f2$ (%)		Maximum Excess of flow (%)		Number of Hubs with flow Excess		Total Number of Hubs in Solution	
	Bi-criteria	Pareto	Bi-criteria	Pareto	Bi-criteria	Pareto	Bi-criteria	Pareto	Bi-criteria	Pareto
20TT	-8.16	-2.83	226.79	-1.42	476.09	612.74	2	1	2	2
	-7.36	-1.80	206.99	-11.23	423.06	147.69	2	1	2	2
	-6.93	-1.93	196.13	-8.14	393.094	134.16	2	1	2	2
	-6.04	-6.04	-19.15	-19.15	82.21	82.21	1	1	1	1
25LT	-4.73	-2.56	-0.41	-1.09	30.56	111.34	1	1	3	2
	-4.71	-0.76	-0.51	-1.22	30.56	108.89	1	1	3	3
	-0.81	-0.81	-4.25	-4.25	21.53	21.53	2	2	3	3
	-0.73	-0.73	-4.38	-4.38	21.53	21.53	2	2	3	3
25TT	-9.98	-9.98	-9.37	-9.37	154.70	154.70	1	1	1	1
	-8.78	-8.56	-10.24	-10.41	74.77	102.39	1	1	2	2
	-1.88	-4.81	-20.57	-11.33	28.10	141.27	1	1	2	3
	-1.51	-5.87	-20.92	-11.19	34.40	386.59	1	2	2	2
40LT	-4.58	-4.58	-1.43	-1.43	108.36	108.36	1	1	2	2
	-4.57	-7.86	-1.51	-56.77	105.66	191.64	1	1	2	2
	-3.08	-0.18	-2.49	-58.63	70.94	156.49	2	2	2	2
	-2.62	-7.74	-2.61	-57.13	66.93	184.72	2	1	2	2
40TL	-1.93	-1.93	-6.78	-6.78	38.48	38.48	1	1	1	1
	2.24	2.24	-10.99	-10.99	32.06	32.06	1	1	1	1
	2.95	7.18	-11.15	-10.59	0.00	0.00	0	0	2	2
	3.44	14.17	-11.52	-21.30	0.00	558.23	0	1	2	1
40TT	-4.17	-2.86	-10.60	-0.92	55.54	26.47	2	1	2	1
	-4.05	-4.05	-11.00	-11.00	61.72	61.72	1	1	2	2
	-0.21	-4.15	-12.94	-10.33	90.99	82.29	1	2	2	2
	-0.04	-13.70	-12.95	7.97	91.20	130.87	1	2	2	2

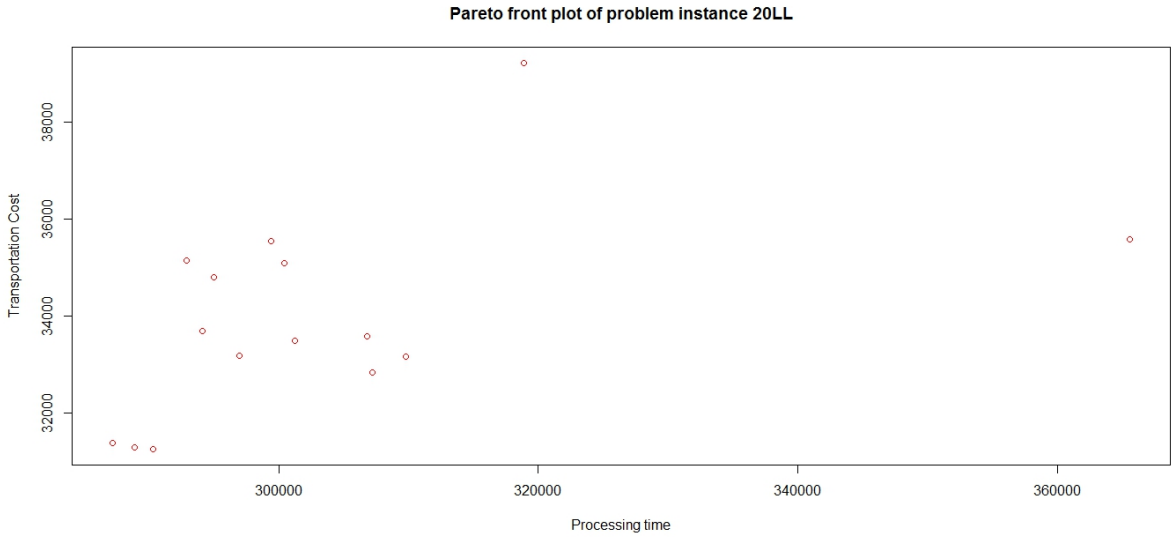

Figure 7.1: Pareto Front plot of problem instance 20LL

7.4.2 Sum of Ranks

In Tables 7.8, 7.9 and 7.10, a comparison of the computed results from the Sum of Ranks fitness evaluation strategy, and that of [12] are presented. The same format used for the results for the Pareto Ranking scheme is adopted here. In the tables, "Summed" represents that of the Sum of Ranks' results.

The same observation and analysis made for the computed results of the Pareto Ranking scheme can be made here. This approach results appear to compete with the Pareto Ranking scheme as the tables clearly depict.

Table 7.8: Group I: Summed Ranks' Results & Comparison with Costa et al. [12]

Problem	$\Delta f1$ (%)		$\Delta f2$ (%)		Maximum Excess of flow (%)		Number of Hubs with flow Excess		Total Number of Hubs in Solution	
	Bi-criteria	Summed	Bi-criteria	Summed	Bi-criteria	Summed	Bi-criteria	Summed	Bi-criteria	Summed
10LT	-10.65	-10.65	520.72	520.72	752.28	752.28	2	2	3	3
	-1.79	3.79	64.83	-31.89	47.94	0.00	1	0	3	2
	1.30	2.01	-6.24	-30.78	0.00	0.00	0	0	3	2
	2.90	3.22	-7.89	3.477	0.00	0.00	0	0	3	2
20LT	-2.72	-2.72	0.85	0.85	15.04	15.04	1	1	2	2
	-1.31	-3.47	0.29	10.99	4.44	23.94	1	1	2	3
	0.47	-6.07	-0.56	11.65	2.95	86.30	1	1	2	2
	8.53	8.53	-6.63	-6.63	82.21	82.21	1	1	1	3
25LL	-0.97	8.92	0.94	-27.86	15.04	27.15	1	1	2	3
	-0.26	5.77	0.19	-29.57	0.76	0.43	1	1	2	2
	2.60	2.60	-18.87	-18.87	0.00	0.00	0	0	2	2
	2.84	9.99	-19.12	-27.99	0.00	0.00	0	0	2	2
25TL	-4.72	-4.72	97.92	97.92	137.34	137.34	1	1	1	2
	12.30	9.23	-9.25	-4.84	0.00	0.00	0	0	2	2
	15.66	0.05	-10.94	-0.04	0.00	0.00	0	0	2	3
	18.76	18.76	-16.20	-16.20	0.00	0.00	0	0	1	1
40LL	-0.40	0.42	296.22	-13.58	534.82	317.71	1	1	2	1
	-0.29	9.26	287.75	-23.01	516.86	558.28	1	0	2	4
	0.07	0.07	-21.33	-21.33	0.00	0.00	0	0	2	2
	0.23	3.92	-21.96	-28.47	0.00	0.00	0	0	2	2

Table 7.9: Group II: Summed Ranks' Results & Comparison with Costa et al. [12]

Problem	$\Delta f1$ (%)		$\Delta f2$ (%)		Maximum Excess of flow (%)		Number of Hubs with flow Excess		Total Number of Hubs in Solution	
	Bi-criteria	Summed	Bi-criteria	Summed	Bi-criteria	Summed	Bi-criteria	Summed	Bi-criteria	Summed
10LL	0.77	9.63	-8.19	-23.27	0.00	0.00	0	0	3	3
	1.58	8.12	-10.31	-17.19	0.00	0.00	0	0	3	2
	2.57	2.57	-18.36	-18.36	0.00	0.00	0	0	2	2
	3.76	3.76	-20.48	-20.48	0.00	0.00	0	0	2	2
10TL	0.43	0.43	-6.03	-6.03	0.00	0.00	0	0	2	2
	2.13	2.13	-7.29	-7.29	0.00	0.00	0	0	2	2
	4.15	4.15	-8.81	-8.81	0.00	0.00	0	0	2	2
	9.55	8.59	-16.81	-8.46	0.00	0.00	0	0	2	2
10TT	1.53	1.53	-1.76	-1.76	0.00	0.00	0	0	3	3
	4.04	9.31	-6.32	-3.17	0.00	0.00	0	0	2	3
	5.23	5.23	-6.50	-7.10	0.00	0.00	0	0	2	2
	7.07	7.07	-6.51	-6.51	0.00	0.00	0	0	2	2
20LL	4.26	4.26	-7.02	-7.02	0.00	0.00	0	0	2	2
	4.60	4.60	-7.46	-7.46	0.00	0.00	0	0	2	2
	6.53	6.53	-11.13	-11.13	0.00	0.00	0	0	2	2
	6.98	6.98	-11.50	-11.50	0.00	0.00	0	0	2	2
20TL	0.39	0.39	-0.42	-0.42	0.00	0.00	0	0	2	2
	1.58	0.00	-0.84	0.00	0.00	0.00	0	0	2	2
	2.60	2.60	-5.65	-5.65	0.00	0.00	0	0	1	1
	15.57	11.53	-10.82	-6.01	0.00	0.00	0	0	2	2

Table 7.10: Group III: Summed Ranks' Results & Comparison with Costa et al. [12]

Problem	$\Delta f1$ (%)		$\Delta f2$ (%)		Maximum Excess of flow (%)		Number of Hubs with flow Excess		Total Number of Hubs in Solution	
	Bi-criteria	Summed	Bi-criteria	Summed	Bi-criteria	Summed	Bi-criteria	Summed	Bi-criteria	Summed
20TT	-8.16	-5.56	226.79	-1.53	476.09	599.00	2	1	2	2
	-7.36	-5.71	206.99	0.07	423.06	806.00	2	1	2	2
	-6.93	-3.23	196.13	-12.57	393.09	146.20	2	1	2	2
	-6.04	-6.04	-19.15	-19.15	82.21	82.21	1	1	1	1
25LT	-4.73	-4.45	-0.41	-1.22	30.56	607.00	1	2	3	2
	-4.71	-7.49	-0.51	2.53	30.56	12.34	1	1	3	2
	-0.81	-0.81	-4.25	-4.25	21.53	21.53	2	2	3	3
	-0.73	-3.72	-4.38	2.40	21.53	101.20	2	1	3	3
25TT	-9.98	-9.98	-9.37	-9.37	154.70	154.70	1	1	1	1
	-8.78	-8.56	-10.24	-10.41	74.77	102.40	1	1	2	2
	-1.88	-1.44	-20.57	-4.72	28.10	79.30	1	1	2	3
	-1.51	-8.35	-20.92	-10.54	34.40	91.20	1	1	2	2
40LT	-4.58	-4.58	-1.43	-1.43	108.36	108.36	1	1	2	2
	-4.57	-1.32	-1.51	-59.82	105.66	114.40	1	1	2	4
	-3.08	-3.05	-2.49	-69.30	70.94	93.60	2	1	2	3
	-2.62	-1.51	-2.61	-59.78	66.93	11.44	2	1	2	4
40TL	-1.93	-1.93	-6.78	-6.78	38.48	38.48	1	1	1	1
	2.24	2.24	-10.99	-10.99	32.06	32.06	1	1	1	1
	2.95	-0.02	-11.15	-0.13	0.00	10.00	0	1	2	2
	3.44	0.18	-11.52	-1.01	0.00	19.50	0	1	2	2
40TT	-4.17	-4.17	-10.60	-10.60	55.54	55.54	2	2	2	2
	-4.05	-15.78	-11.00	16.63	61.72	180.80	1	1	2	2
	-0.21	-0.30	-12.94	14.14	90.99	11.74	1	1	2	4

7.4.3 Weighted Sum

In Tables 7.11, 7.12 and 7.13, a comparison of our computed results of the weighted sum fitness evaluation strategy, and that of the work of [12]. Each problem is presented with one solution since this approach returns only one solution at the end of each run. In these tables, "Weighted" represents the results of the Weighted Sum.

Table 7.11: Group I: Weighted Sum's Results & Comparison with Costa et al. [12]

Problem	$\Delta f1$ (%)		$\Delta f2$ (%)		Maximum Excess of flow (%)		Number of Hubs with flow Excess		Total Number of Hubs in Solution	
	Bi-criteria	Weighted	Bi-criteria	Weighted	Bi-criteria	Weighted	Bi-criteria	Weighted	Bi-criteria	Weighted
10LT	-10.65	-1.79	520.72	14.01	752.28	308.56	2	1	3	3
	-1.79		64.83		47.94		1		3	
	1.30		-6.24		0.00		0		3	
	2.90		-7.89		0.00		0		3	
20LT	-2.72	-4.39	0.85	23.65	15.04	55.52	1	1	2	3
	-1.31		0.29		4.44		1		2	
	0.47		-0.56		2.95		1		2	
	8.53		-6.63		82.21		1		1	
25LL	-0.97	-0.97	0.94	0.94	15.04	15.04	1	1	2	2
	-0.26		0.19		0.76		1		2	
	2.60		-18.87		0.00		0		2	
	2.84		-19.12		0.00		0		2	
25TL	-4.72	-4.72	97.92	97.92	137.34	137.34	1	1	1	2
	12.30		-9.25		0.00		0		2	
	15.66		-10.94		0.00		0		2	
	18.76		-16.20		0.00		0		1	
40LL	-0.40		296.22		534.82		1		2	
	-0.29		287.75		516.86		1		2	
	0.07	0.09	-21.33	-21.33	0.00	0.00	0	0	2	2
	0.23		-21.96		0.00		0		2	

Table 7.12: Group II: Weighted Sum's Results & Comparison with Costa et al. [12]

Problem	$\Delta f1$ (%)		$\Delta f2$ (%)		Maximum Excess of flow (%)		Number of Hubs with flow Excess		Total Number of Hubs in Solution	
	Bi-criteria	Weighted	Bi-criteria	Weighted	Bi-criteria	Weighted	Bi-criteria	Weighted	Bi-criteria	Weighted
10LL	0.77		-8.19		0.00		0		3	
	1.58		-10.31		0.00		0		3	
	2.57	2.57	-18.36	-18.36	0.00	0.00	0	0	2	2
	3.76		-20.48		0.00		0		2	
10TL	0.43	0.43	-6.03	-6.03	0.00	0.00	0	0	2	2
	2.13		-7.29		0.00		0		2	
	4.15		-8.81		0.00		0		2	
	9.55		-16.81		0.00		0		2	
10TT	1.53	1.53	-1.76	-1.76	0.00	0.00	0	0	3	3
	4.04		-6.32		0.00		0		2	
	5.23		-6.50		0.00		0		2	
	7.07		-6.51		0.00		0		2	
20LL	4.26	2.02	-7.02	22.63	0.00	0.00	0	0	2	3
	4.60		-7.46		0.00		0		2	
	6.53		-11.13		0.00		0		2	
	6.98		-11.50		0.00		0		2	
20TL	0.39		-0.42		0.00		0		2	
	1.58		-0.84		0.00		0		2	
	2.60	2.60	-5.65	-5.65	0.00	0.00	0	0	1	1
	15.57		-10.82		0.00		0		2	

Table 7.13: Group III: Weighted Sum's Results & Comparison with Costa et al. [12]

Problem	$\Delta f1$ (%)		$\Delta f2$ (%)		Maximum Excess of flow (%)		Number of Hubs with flow Excess		Total Number of Hubs in Solution	
	Bi-criteria	Weighted	Bi-criteria	Weighted	Bi-criteria	Weighteded	Bi-criteria	Weighted	Bi-criteria	Weighted
20TT	-8.16	-6.04	226.79	-19.15	476.09	599.00	2	1	2	2
	-7.36		206.99		423.06		2		2	
	-6.93		196.13		393.09		2		2	
	-6.04		-19.15		82.21		1		1	
25LT	-4.73	-7.99	-0.41	59.62	30.56	174.84	1	1	3	3
	-4.71		-0.51		30.56		1		3	
	-0.81		-4.25		21.53		2		3	
	-0.73		-4.38		21.53		2		3	
25TT	-9.98	-9.98	-9.37	-9.37	154.70	154.70	1	1	1	1
	-8.78		-10.24		74.77		1		2	
	-1.88		-20.57		28.10		1		2	
	-1.51		-20.92		34.40		1		2	
40LT	-4.58	-3.77	-1.43	-7.67	108.36	172.54	1	2	2	4
	-4.57		-1.51		105.66		1		2	
	-3.08		-2.49		70.94		2		2	
	-2.62		-2.61		66.93		2		2	
40TL	-1.93	-1.93	-6.78	-6.78	38.48	38.48	1	1	1	1
	2.24		-10.99		32.06		1		1	
	2.95		-11.15		0.00		0		2	
	3.44		-11.52		0.00		0		2	
40TT	-4.17	-4.17	-10.60	-10.60	55.54	55.54	2	2	2	2
	-4.05		-11.00		61.72		1		2	
	-0.21		-12.94		90.99		1		2	
	-0.04		-12.95		91.20		1		2	

7.4.4 Analysis of Results of the Fitness Evaluation Strategies

The results of the Pareto Ranking fitness evaluation strategy, shown in Tables 7.5, 7.6 and 7.7, reveal that the strategy provides more alternative solutions that have not been provided in [12]. The same observation can be made of the results from the Sum of Ranks approach. Also, from Tables 7.11, 7.12, and 7.13, it can be observed that the Weighted Sum approach to the fitness evaluation of the MOGA has been effective and its results compete with that of [12]. Furthermore, MOGA either improved on the values from [12] or obtained solutions that are within the range of solutions provided by them. The results show that, all the three fitness evaluation strategies are effective and suited for applying to the multi-objective approach to the CSAHLP.

7.5 Comparison of the Fitness Evaluation Strategies with [12]

In Tables 7.14, 7.15 and 7.16, a comparison of the computed results of the three fitness evaluation strategies are provided. From these computed results, it can be observed that MOGA performed effectively well as compared to [12]. As a result, the MOGA modelling of the CSAHLP by removing the capacity constraints while introducing a second objective measure as done by [12], presents compelling results that give the DM several options to choose from instead of the previous works that always present two options to the DM: acceptance/rejection of solutions from the SOGA.

The fitness evaluation strategies employed deserve some attention. A careful look at the results show that generally, all strategies performed fairly the same although the weighted sum was effective in finding all the existing solutions by Costa et al. [12].

7.6 Conclusion

In conclusion, it is clear from the computed results and the analysis made so far, that the MOGA approach to the CSAHLP is most suitable as compared to the SOGA approach. The approach gives more power to the DM and does not tie him/her to just an option of rejecting or accepting a given solution to a problem instance. However, he/she is provided with a list of alternative solutions, and more details on the consequence of favouring one objective at the expense of the other. The results of

Table 7.14: Group I: Comparison of MOGA with Costa et al. [12]

Problem	Costa et al. [12] Trade-off(%)		Pareto Ranks Trade-off(%)		Sum Ranks Trade-off(%)		Weighted Trade-off(%)	
	Cost	Time	Cost	Time	Cost	Time	Cost	Time
10LT	-10.65	520.72	10.82	-32.55	-10.65	520.72	-1.79	14.01
	-1.79	64.83	-1.79	64.83	3.79	-31.89		
	1.30	-6.24	6.50	-31.2	2.01	-30.78		
	2.90	-7.89	3.22	3.47	3.22	3.47		
20LT	-2.72	0.85	-2.72	0.85	-2.72	0.85	-4.39	23.65
	-1.31	0.29	-6.55	237.35	-3.47	10.99		
	0.47	-0.56	-1.29	0.57	-6.06	11.65		
	8.83	-6.33	8.83	-6.33	8.83	-6.33		
25LL	-0.97	0.94	18.75	-34.17	8.92	-27.86	0.97	0.94
	-0.26	0.19	17.80	-28.10	5.77	-29.57		
	2.60	-18.87	2.60	-18.87	2.60	-18.87		
	2.84	-19.12	4.34	-17.09	9.98	-27.99		
25TL	-4.72	97.92	-4.72	97.92	-4.72	97.92	-4.72	97.92
	12.30	-9.25	18.94	-1.14	9.23	-4.84		
	15.66	-10.94	14.38	2.25	0.05	-0.03		
	18.76	-16.20	19.39	-1.47	18.76	-16.20		
40LL	-0.40	296.22	5.75	-19.98	0.42	-13.58		
	-0.29	287.75	4.46	-22.97	9.26	-23.01		
	0.07	-21.33	0.07	-21.33	0.07	-21.33	0.09	-21.33
	0.23	-21.96	2.57	-5.10	3.92	-28.47		

MOGA also compete with that of Costa et al. [12] and includes some new solutions that were not reported by their works.

Table 7.15: Group II: Comparison of MOGA with Costa et al. [12]

Problem	Costa et al. [12] Trade-off(%)		Pareto Ranks Trade-off(%)		Sum Ranks Trade-off(%)		Weighted Sum Trade-off(%)	
	Cost	Time	Cost	Time	Cost	Time	Cost	Time
10LL	0.77	-8.19	6.85	-16.68	9.62	-23.26		
	1.58	-10.31	1.58	-10.31	8.12	-17.19		
	2.57	-18.36	2.57	-18.36	2.57	-18.36	2.57	-18.36
	3.76	-7.89	3.76	-7.89	3.76	-7.89		
10TL	0.43	-6.03	0.43	-6.03	0.43	-6.03	0.43	-6.03
	2.13	-7.29	2.13	-7.29	2.13	-7.29		
	4.15	-8.81	12.50	-21.60	7.27	-6.26		
	9.55	-16.81	8.59	-8.46	8.59	-8.46		
10TT	1.53	-1.76	1.53	-1.76	1.53	-1.76	1.53	-1.76
	4.04	-6.32	9.31	-3.17	-6.32	-7.24		
	5.23	-6.50	5.23	-7.40	5.23	-7.40		
	7.07	-6.51	7.07	-6.51	5.94	-2.21		
20LL	4.26	-7.02	4.26	-7.02	4.26	-7.02		
	4.60	-7.46	4.60	-7.46	4.60	-7.46	20.2	22.63
	6.53	-11.13	6.53	-11.13	6.53	-11.13		
	6.98	-11.50	6.98	-11.50	6.98	-11.50		
20TL	0.39	-0.42	0.39	-0.42	0.39	-0.42		
	1.58	-0.84	2.81	-0.74	0.00	0.00		
	2.60	-5.65	2.60	-5.65	2.60	-5.65	2.60	-5.65
	15.57	-10.82	16.39	-11.39	11.53	-6.01		

Table 7.16: Group III: Comparison of MOGA with Costa et al. [12]

Problem	Costa et al. [12] Trade-off(%)		Pareto Ranks Trade-off(%)		Sum Ranks Trade-off(%)		Weighted Sum Trade-off(%)	
	Cost	Time	Cost	Time	Cost	Time	Cost	Time
20TT	-8.16	226.79	-2.83	-1.42	-5.56	-1.53	-6.04	-19.15
	-7.36	206.99	-1.80	-11.23	-5.71	0.07		
	-6.93	196.13	-1.93	-8.14	-3.23	-12.57		
	-6.04	-19.15	-6.04	-19.15	-6.04	-19.15		
25LT	-4.73	-0.41	-2.56	-1.09	-4.44	-1.22	-7.99	59.62
	-4.71	-0.51	-0.76	-1.22	24.61	-5.66		
	-0.81	-4.25	-0.81	-4.25	-0.81	-4.25		
	-0.73	-4.38	-0.73	-4.38	-3.72	2.40		
25TT	-9.98	-9.37	-9.98	-9.37	-9.98	-9.37	-9.98	-9.37
	-8.78	-10.24	-8.56	-10.41	-8.56	-10.41		
	-1.88	-20.57	-4.81	-11.33	-1.44	-4.715		
	-1.51	-20.92	-5.87	-11.19	-8.35	-10.54		
40LT	-4.58	-1.43	-4.58	-1.43	-4.58	-1.43	-3.77	-7.67
	-4.57	-1.51	-7.86	-56.77	-1.32	-59.82		
	-3.08	-2.49	-0.18	-58.63	-3.05	-69.30		
	-2.62	-2.61	-7.74	-57.13	-1.51	-59.78		
40TL	-1.93	-6.78	-1.93	-6.78	-1.93	-6.78	-1.93	-6.78
	2.24	-10.99	2.24	-10.99	2.24	-10.99		
	2.95	-11.15	7.18	-10.59	-0.02	-0.13		
	3.44	-11.52	14.16	-21.30	0.18	-1.01		
40TT	-4.17	-10.60	-2.86	-0.92	-4.17	-10.60	-4.17	-10.60
	-4.05	-11.00	-4.05	-11.00	-15.78	-15.78		
	-0.21	-12.94	-4.14	-10.32	-0.29	14.14		
	-0.04	-12.95	-13.69	7.97	-17.39	22.57		

Chapter 8

Conclusion & Future Works

SAHLP is a NP-hard combinatorial optimization problem with many practical applications. This thesis studied both the capacitated and un-capacitated version of the SAHLP and proposed a GA solution for solving the problem approximately. Although previous work is found in the literature using GA, the GA work is limited especially for the capacitated version. Furthermore, to the authors knowledge, this is the first multi-objective GA for this problem. Two new crossover operators were introduced where one of them was shown to outperform existing ones in solution quality, while the second one is competitive with existing crossover operators.

The experimental results show the effectiveness of the proposed GA for larger AP data set of node sizes 300 and 400 which have only been recently introduced. Besides easily finding the currently known best solutions for the problem instances, the GA also introduced new best solutions for some problem instances. A multi-objective GA approach is proposed for the CSAHLP by removing the capacity constraints, and introducing the time to process flow as the second objective measure in addition to the network transportation cost as the first objective. Doing so provides the decision maker with more options of solutions to choose from, instead of the usual "accept" or "reject" options that the single objective approach has always presented. The results of the multi-objective GA are competitive with that of previous multi-objective approach based on Integer Programming and also provide new solutions.

The following are areas that should be considered for future work:

- Given the good solutions provided by the MOGA, applying it to larger problem instances should be done.
- Extend the proposed GA approach to similar hub location problems such as the multi-hub and dynamic versions of the HLP.

- Perform a fitness landscape analysis for the problem.

Bibliography

[1] S. Abdinnour-Helm. A hybrid heuristic for the un-capacitated hub location problem. *European Journal of Operational Research*, 106:489–499, 1998.

[2] S. Abdinnour-Helm. Using simulated annealing to solve the p-hub median problem. *International Journal of Physical Distribution and Logistics Management*, 31:209–237, 2001.

[3] S. Abdinnour-Helm and M. A. Venkataramanan. Solution approaches to hub location problems. *Annals of Operations Research*, 78:31–50, 1998.

[4] T. Aykin. Lagrangian relaxation based approaches to capacitated hub-and-spoke net- work design problem. *European Journal of Operational Research*, 79:501–23, 1994.

[5] T. Aykin. Networking policies for hub-and-spoke systems with application to the air transportation system. *Transportation Science*, 29:201 221, 8 1995.

[6] A. Bailey, B. Ombuki-Berman, and S. Asobiela. Discrete pso for the uncapacitated single allocation hub location problem. *IEEE Symposium Series on Computational Intelligence, Singapore*, 2013.

[7] P. Bentley and J. Wakefield. Finding acceptable solutions in the pareto-optimal range using multiobjective genetic algorithms. *Soft Computing in Engineering Design and Manufacturing*, 1997.

[8] S. Bergen and B. Ross. Evolutionary art using summed multi-objective ranks. *Genetic Programming - Theory and Practice VIII*, May 2010.

[9] Jeng-Fung Chen. Ahybrid heuristic for the uncapacitated single allocation hub location problem. *The International Journal of Management Science*, 35:211 220, 2007.

[10] I. Contreras, J. Cordeau, and G. Laporte. The dynamic uncapacitated hub location problem. *Transportation Science*, 45(1):18–32, 9 2009.

[11] I. Contreras, J. A. Diaz, and E. Fernandez. Lagrangean relaxation for the capacitated hub location problem with single assignment. *Operations Research Spectrum*, 31:483505, 2009.

[12] M.G. Costa, M. E. Captivo, and J. Climaco. Capacitated single allocation hub location problem - a bi-criteria approach. *Computers & Operations Research*, 35: 3671–3695, 2008.

[13] de Camargo, R. S., G. Miranda, Jr., R. P. Ferreira, and H. P. Luna. Multiple allocation hub-and-spoke network design under hub congestion. *Computers & Operations Research*, 36(12):3097–3106, 2009.

[14] A. T. Ernst and M. Krishnamoorthy. An exact solution approach based on shortest paths for p-hub median problems. *INFORMS Journal on Computing*, 10(2):149–62, 1996.

[15] A. T. Ernst and M. Krishnamoorthy. Efficient algorithms for the uncapacitated single allocation p-hub median problem. *Location Science*, 4(3):13954., 1996.

[16] A. T. Ernst and M. Krishnamoorthy. Exact and heuristic algorithms for the uncapacitated multiple allocation p-hub median problem. *European Journal of Operational Research*, 104:10012., 1998.

[17] A. T. Ernst and M. Krishnamoorthy. Solution algorithms for the capacitated single allocation hub location problem. *Annals of Operations Research*, 86:141–159, 1999.

[18] V. Filipovic, J. Kratica, D. Tosic, and D. Dugosija. GA inspired heuristic for uncapacitated single allocation hub location problem. *Application of Soft Computing, AISC*, 58:149–158, 2009.

[19] D. E. Goldberg. *Genetic Algorithm in search, optimisation and machine learning.* MA: Addison-Wesley, 1989.

[20] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence.* Michigan: University of Michigan Press, 1975.

[21] J. G. Klincewicz. Hub location in backbone tributary network design: a review. *Computers & Operations Research*, 6:307–335, 1998.

[22] J. Kratica, Z. Stanimirovic, D. Tosic, and V. Filipovic. Two genetic algorithms for solving the uncapacitated single allocation p-hub median problem. *European Journal of Operational Research*, 182(1):15–28, 2007.

[23] M. Mitchell. *An Introduction to Genetic algorithm.* Cambridge, MA: MIT, 1998.

[24] M. Naeem. Using genetic algorithms for the single allocation hub location problem. Master's thesis, Brock University, St. Catharines, Ontario, Canada, 2009.

[25] M. Naeem and B. M. Ombuki-Berman. An effficient genetic algorithm for the uncapacitated single allocation hub location problem. *IEEE Congress on Evolutionary Computation*, pages 1–8, 2010.

[26] M. E. O'kelly. A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research*, 32:393–404, 1987.

[27] B. Ombuki, B. J. Ross, and F. Hanshar. Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, 24:17–30, 2006.

[28] B. M. Ombuki-Berman, A. Runka, and F. T. Hanshar. Waste collection vehicle routing problem with time windows using multi-objective genetic algorithms. *From Proceeding (574) Computational Intelligence*, 574, 2007.

[29] M. Randall. Solution approaches for the capacitated single allocation hub location problem using ant colony optimisation. *Computational Optimization and Applications: An International Journal*, 39:239–261, 1998.

[30] K. B. A. Sibel. Network hub location problem: State of the art. *European Journal of Operational Research*, 190:44758, 2001.

[31] M. R. Silva and C. B. Cunha. New simple and efficient heuristics for the uncapacitated singleal location hub location problem. *Computers & Operations Research*, 36:3152 – 3165, 2009.

[32] D. Skorin-Kapov, J. Skorin-Kapov, and M. E. O'Kelly. Tight linear programming relaxations of uncapacitated p-hub median problems. *European Journal of Operational Research*, 94:58293, 1996.

[33] Z. Stanimirovic. *Solving the Capacitated Single Allocation Hub Location Problem Using Genetic Algorithm*, chapter 55, pages 464–471. 2007.

[34] Z. Stanimirovic. A genetic algorithm approach for the capacitated single allocation p-hub median problem. *Computing and Informatics*, 29(1):117–132, 2010.

[35] G. D. Taylor, S. Harit, and J. R. English. Hub and spoke networks in truckload trucking: Configuration and operational concerns. *Logistics and Transportation*, 31:209–237, 1995.

[36] H. Topcuoglu, F. Corut, M. Ermis, and G. Yilmaz. Solving the uncapacitated hub location problem using genetic algorithms. *Computers & Operations Research*, 32:967–984, 2005.

[37] D. A. V. Veldhuizen and G. B. Lamont. Multiobjective evolutionary algorithm: Analyzing the state-of-the-art. *Evolutionary Computation*, 8(2):125–147, 2000.

Appendix A

Appendix of SOGA Average Results

Tables A.1, A.2, A.3, and A.4 show the averages of 50 runs for both the uncapacitated and capacitated AP data instances using the SOGA. In these tables, the problem and its known best are represented in the first two columns, and the next columns are the averages of 50 runs for each problem instance obtained using each of the four crossover type implementation: GA-BCRC, GA-RESC, GA-DCEC, and GA-MCEC respectively. The values in bold figure represent the best average among the four GA implementations for the given problem instance.

Table A.1: USAHLP: Average of 50 Runs

Problem	Known Best	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
10L	224250.05	**228908.91**	244124.75	235709.06	235117.40
20L	234690.95	250082.56	248927.33	**236308.11**	237018.49
25L	236650.62	266384.41	251772.24	**239605.69**	240011.56
40L	240986.23	273365.26	254978.64	243920.77	**242890.98**
50L	237421.98	264429.46	251776.90	**238789.13**	240839.83
100L	238016.28	265045.02	249619.93	**238844.38**	240313.31
200L	233802.97	255428.19	250791.39	237211.85	**235201.73**
300L	264,837.88	276976.78	283228.13	**275382.32**	275861.76
400L	268,164.13	282890.09	288406.22	279808.70	**278838.74**
10T	263399.94	**265986.17**	274056.35	270725.21	270428.68
20T	271128.18	293090.86	277907.27	**277463.76**	277745.34
25T	295667.84	345485.56	**295667.84**	**295667.84**	**295667.84**
40T	293164.83	335040.55	**293164.84**	**293164.84**	**293164.84**
50T	300420.98	350450.44	**300420.99**	**300420.99**	**300420.99**
100T	305097.96	352505.05	**305097.95**	**305097.95**	**305097.95**
200T	272188.11	294382.25	294192.25	279848.29	**279394.09**
300T	276,047.75	**298487.60**	305999.61	304232.93	303846.66
400T	284,212.47	304079.04	**301497.04**	301941.64	301941.64

Table A.2: Small Problem Instances of CSAHLP: Averages of 50 Runs

Problem	Known Best [11]	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
10LL	224250.05	**227246.38**	239920.81	232268.60	237872.27
10LT	250992.26	256592.55	256253.31	**252095.89**	255557.32
10TL	263399.94	**265537.11**	273381.50	271886.28	271202.10
10TT	263399.94	264825.89	265162.81	**264452.45**	264521.09
20LL	234690.94	252474.86	241770.42	**234690.96**	237535.49
20LT	253517.40	269430.21	253630.48	253726.48	**253517.40**
20TL	271128.18	292975.98	278167.72	277656.24	**277477.56**
20TT	296035.40	313091.94	300430.06	298713.13	**297815.47**
25LL	238977.95	262161.25	**240664.96**	241567.15	240957.55
25LT	276372.50	290992.20	282519.60	280250.63	**279572.30**
25TL	310317.64	354190.86	317842.88	315771.09	**313231.02**
25TT	348369.15	367621.69	357694.32	355511.70	**353480.83**
40LL	241955.71	273983.08	**242065.46**	242136.18	242133.53
40LT	272218.32	290925.54	280480.97	276594.07	**274646.01**
40TL	298919.01	334567.14	302204.16	**301387.81**	301880.38
40TT	354874.10	384965.20	375508.43	366848.57	**364702.35**
50LL	238520.59	265522.72	**238520.59**	**238520.59**	textbf238520.59
50LT	272897.49	298174.33	288895.21	285785.20	**284421.55**
50TL	319015.77	346652.02	327214.61	**325489.86**	326084.12
50TT	417440.99	451616.64	428285.13	425345.02	**425075.82**

Table A.3: Medium Problem Instances of CSAHLP: Averages of 50 Runs

Problem	Known Best [11]	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
60LL	225917.21	247387.69	**225917.21**	225921.72	225928.26
60LT	253761.98	289531.95	267318.33	**263974.39**	266303.77
60TL	252496.66	290602.52	254910.83	**254276.47**	254318.00
60TT	351274.72	383965.19	357357.47	**354521.22**	355335.19
70LL	236817.35	256803.83	238659.92	**238302.64**	239191.99
70LT	257454.36	277224.83	265600.10	**260790.91**	261783.32
70TL	271283.82	327439.31	272606.31	**272523.48**	272609.90
70TT	387380.20	458490.13	408686.96	393717.97	**392438.42**
75LL	238024.22	262337.93	239275.32	**238309.59**	238339.83
75LT	256188.12	295613.70	264718.90	**256971.72**	258178.16
75TL	303363.55	333721.43	**307697.86**	308552.14	307931.85
75TT	347189.81	388920.03	**362213.62**	364118.37	363087.62
90LL	224195.72	261832.99	224395.72	**224195.72**	**224195.72**
90LT	246026.24	295875.02	253126.52	252877.87	**250520.30**
90TL	281561.56	317820.56	**301351.42**	303011.49	302133.99
90TT	337008.93	374150.02	362693.19	371230.92	**358341.33**

Table A.4: Large Problem Instances: Averages of 50 Runs

Problem	Known Best [11]	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
100LL	246713.97	269365.00	254230.87	255739.71	**252928.00**
100LT	256155.33	281269.03	269311.50	279421.16	**261433.00**
100TL	362950.09	394053.05	**381088.07**	384143.42	381939.35
100TT	474680.32	503155.44	520497.94	535997.77	**502402.95**
125LL	239920.75	263892.30	**242644.72**	243050.25	242813.88
125LT	251259.16	292229.84	264260.70	267390.67	**262569.51**
125TL	246486.69	295335.09	**246486.69**	246672.95	246854.20
125TT	291807.35	354176.29	331973.39	327513.89	**325195.20**
150LL	234765.44	261645.47	**234883.92**	235227.98	235249.10
150LT	250186.53	301295.68	263219.10	266603.67	**258391.64**
150TL	262822.24	286917.48	270994.89	**267835.56**	268662.81
150TT	323992.37	373242.57	341081.06	342828.77	**340008.72**
175LL	227997.58	255993.23	229666.05	**229032.14**	229851.88
175LT	251540.80	296435.24	269968.89	267574.88	**263995.86**
175TL	244860.41	278593.53	**246138.60**	249265.05	247222.78
175TT	308310.13	371058.54	321185.31	324735.79	**319623.93**
200LL	231069.50	272300.62	243752.80	245603.14	**242915.52**
200LT	268820.57	324258.53	283453.45	288371.53	**277419.59**
200TL	273443.81	296552.83	**286671.89**	298633.61	286921.45
200TT	290841.84	336251.52	326431.67	343370.82	**305532.15**

Appendix B

Appendix of SOGA Results using CAB Data Set

Tables B.1, B.2, B.3, and B.4 show the results obtained when SOGA was applied on the CAB data set. In these tables, the first two columns represent the α and the f values of the problem instance. The next columns represent the problem's currently known best (Optimal Cost), and the best solution obtained by each of the four crossover implementation types. Where the proposed GAs found the currently known best solution, it is indicated with a $\surd$, and the frequency of obtaining this known best in 50 runs is shown in the brackets next to the $\surd$.

Table B.1: Comparison with Known Best, $n = 10$, CAB Data

α	f	Optimal Cost [31]	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
0.2	100	791.93	$\surd^{(15)}$	$\surd^{(13)}$	$\surd^{(10)}$	$\surd^{(6)}$
	150	915.99	$\surd^{(36)}$	$\surd^{(24)}$	$\surd^{(21)}$	$\surd^{(26)}$
	200	1015.99	$\surd^{(43)}$	$\surd^{(30)}$	$\surd^{(23)}$	$\surd^{(20)}$
	250	1115.99	$\surd^{(38)}$	$\surd^{(12)}$	$\surd^{(21)}$	$\surd^{(20)}$
0.4	100	867.91	$\surd^{(11)}$	$\surd^{(22)}$	$\surd^{(23)}$	$\surd^{(26)}$
	150	974.30	$\surd^{(42)}$	$\surd^{(30)}$	$\surd^{(24)}$	$\surd^{(23)}$
	200	1074.30	$\surd^{(45)}$	$\surd^{(25)}$	$\surd^{(23)}$	$\surd^{(17)}$
	250	1174.30	$\surd^{(44)}$	$\surd^{(10)}$	$\surd^{(10)}$	$\surd^{(10)}$
0.6	100	932.62	$\surd^{(40)}$	$\surd^{(20)}$	$\surd^{(20)}$	$\surd^{(23)}$
	150	1032.62	$\surd^{(40)}$	$\surd^{(20)}$	$\surd^{(15)}$	$\surd^{(17)}$
	200	1131.05	$\surd^{(2)}$	$\surd^{(9)}$	$\surd^{(30)}$	$\surd^{(30)}$
	250	1181.05	$\surd^{(5)}$	$\surd^{(13)}$	$\surd^{(30)}$	$\surd^{(30)}$
0.8	100	990.94	$\surd^{(19)}$	$\surd^{(20)}$	$\surd^{(7)}$	$\surd^{(18)}$
	150	1081.05	$\surd^{(5)}$	$\surd^{(5)}$	$\surd^{(30)}$	$\surd^{(30)}$
	200	1131.05	$\surd^{(6)}$	$\surd^{(7)}$	$\surd^{(30)}$	$\surd^{(30)}$
	250	1181.05	$\surd^{(8)}$	$\surd^{(14)}$	$\surd^{(30)}$	$\surd^{(30)}$
1.0	100	1031.05	$\surd^{(28)}$	$\surd^{(35)}$	$\surd^{(30)}$	$\surd^{(30)}$
	150	1081.05	$\surd^{(29)}$	$\surd^{(34)}$	$\surd^{(30)}$	$\surd^{(30)}$
	200	1131.05	$\surd^{(24)}$	$\surd^{(29)}$	$\surd^{(30)}$	$\surd^{(30)}$
	250	1181.05	$\surd^{(30)}$	$\surd^{(35)}$	$\surd^{(30)}$	$\surd^{(30)}$

Table B.2: Comparison with Known Best, $n = 15$, CAB Data

α	f	Optimal Cost [31]	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
0.2	100	1030.07	$\surd^{(4)}$	$\surd^{(3)}$	$\surd^{(1)}$	1110.09
	150	1239.77	$\surd^{(19)}$	$\surd^{(9)}$	$\surd^{(1)}$	$\surd^{(4)}$
	200	1381.28	$\surd^{(3)}$	$\surd^{(2)}$	$\surd^{(5)}$	$\surd^{(6)}$
	250	1481.28	$\surd^{(6)}$	$\surd^{(3)}$	$\surd^{(1)}$	$\surd^{(3)}$
0.4	100	1179.71	$\surd^{(2)}$	$\surd^{(3)}$	$\surd^{(4)}$	$\surd^{(3)}$
	150	1355.09	$\surd^{(9)}$	$\surd^{(7)}$	$\surd^{(3)}$	$\surd^{(1)}$
	200	1462.62	$\surd^{(43)}$	$\surd^{(25)}$	$\surd^{(2)}$	$\surd^{(1)}$
	250	1556.66	$\surd^{(9)}$	$\surd^{(15)}$	$\surd^{(30)}$	$\surd^{(30)}$
0.6	100	1309.92	$\surd^{(7)}$	$\surd^{(3)}$	$\surd^{(1)}$	$\surd^{(1)}$
	150	1443.97	$\surd^{(21)}$	$\surd^{(33)}$	$\surd^{(30)}$	$\surd^{(30)}$
	200	1506.66	$\surd^{(3)}$	$\surd^{(6)}$	$\surd^{(30)}$	$\surd^{(30)}$
	250	1556.66	$\surd^{(2)}$	$\surd^{(12)}$	$\surd^{(30)}$	$\surd^{(30)}$
0.8	100	1390.76	$\surd^{(22)}$	$\surd^{(19)}$	$\surd^{(3)}$	$\surd^{(1)}$
	150	1456.66	$\surd^{(5)}$	$\surd^{(5)}$	$\surd^{(30)}$	$\surd^{(30)}$
	200	1506.66	$\surd^{(3)}$	$\surd^{(1)}$	$\surd^{(30)}$	$\surd^{(30)}$
	250	1556.66	$\surd^{(4)}$	$\surd^{(5)}$	$\surd^{(30)}$	$\surd^{(30)}$
1.0	100	1406.66	$\surd^{(5)}$	$\surd^{(20)}$	$\surd^{(30)}$	$\surd^{(30)}$
	150	1456.66	$\surd^{(4)}$	$\surd^{(7)}$	$\surd^{(30)}$	$\surd^{(30)}$
	200	1506.66	$\surd^{(4)}$	$\surd^{(8)}$	$\surd^{(30)}$	$\surd^{(30)}$
	250	1556.66	$\surd^{(4)}$	$\surd^{(3)}$	$\surd^{(1)}$	$\surd^{(30)}$

Table B.3: Comparison with Known Best, $n = 20$, CAB Data

α	f	Optimal Cost [31]	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
0.2	100	967.74	$\surd^{(14)}$	$\surd^{(20)}$	$\surd^{(2)}$	$\surd^{(5)}$
	150	1174.53	$\surd^{(5)}$	$\surd^{(9)}$	$\surd^{(10)}$	$\surd^{(10)}$
	200	1324.53	$\surd^{(5)}$	$\surd^{(2)}$	$\surd^{(4)}$	$\surd^{(2)}$
	250	1474.53	$\surd^{(14)}$	$\surd^{(8)}$	$\surd^{(9)}$	$\surd^{(9)}$
0.4	100	1127.09	$\surd^{(4)}$	$\surd^{(11)}$	$\surd^{(4)}$	$\surd^{(2)}$
	150	1297.76	$\surd^{(7)}$	$\surd^{(8)}$	$\surd^{(1)}$	$\surd^{(1)}$
	200	1442.56	$\surd^{(11)}$	$\surd^{(15)}$	$\surd^{(8)}$	$\surd^{(8)}$
	250	1570.91	$\surd^{(12)}$	$\surd^{(7)}$	$\surd^{(1)}$	$\surd^{(4)}$
0.6	100	1269.15	$\surd^{(4)}$	$\surd^{(2)}$	$\surd^{(2)}$	$\surd^{(2)}$
	150	1406.04	$\surd^{(17)}$	$\surd^{(20)}$	$\surd^{(15)}$	$\surd^{(9)}$
	200	1506.04	$\surd^{(15)}$	$\surd^{(2)}$	$\surd^{(6)}$	$\surd^{(4)}$
	250	1701.20	$\surd^{(13)}$	$\surd^{(12)}$	$\surd^{(30)}$	$\surd^{(30)}$
0.8	100	1369.52	$\surd^{(7)}$	$\surd^{(2)}$	$\surd^{(9)}$	$\surd^{(8)}$
	150	1469.52	$\surd^{(24)}$	$\surd^{(24)}$	$\surd^{(3)}$	$\surd^{(2)}$
	200	1520.91	$\surd^{(20)}$	$\surd^{(25)}$	$\surd^{(30)}$	$\surd^{(30)}$
	250	1570.91	$\surd^{(1)}$	$\surd^{5})$	$\surd^{(30)}$	$\surd^{(30)}$
1.0	100	1410.07	$\surd^{(23)}$	$\surd^{(22)}$	$\surd^{(3)}$	$\surd^{(4)}$
	150	1470.91	$\surd^{(5)}$	$\surd^{(12)}$	$\surd^{(30)}$	$\surd^{(30)}$
	200	1520.91	$\surd^{(4)}$	$\surd^{(2)}$	$\surd^{(30)}$	$\surd^{(30)}$
	250	1570.91	$\surd^{(5)}$	$\surd^{(9)}$	$\surd^{(30)}$	$\surd^{(30)}$

Table B.4: Comparison with Known Best, $n = 25$, CAB Data

α	f	Optimal Cost [31]	GA-BCRC	GA-RESC	GA-DCEC	GA-MCEC
0.2	100	1029.63	$\surd^{(3)}$	$\surd^{(2)}$	$\surd^{(1)}$	$\surd^{(1)}$
	150	1217.34	$\surd^{(3)}$	$\surd^{(3)}$	$\surd^{(4)}$	$\surd^{(2)}$
	200	1367.34	$\surd^{(12)}$	$\surd^{(10)}$	$\surd^{(2)}$	$\surd^{(2)}$
	250	1500.90	$\surd^{(5)}$	$\surd^{(12)}$	$\surd^{(18)}$	$\surd^{(10)}$
0.4	100	1187.51	$\surd^{(4)}$	$\surd^{(6)}$	$\surd^{(11)}$	$\surd^{(1)}$
	150	1351.69	$\surd^{(15)}$	$\surd^{(19)}$	$\surd^{(2)}$	$\surd^{(2)}$
	200	1501.62	$\surd^{(15)}$	$\surd^{(11)}$	$\surd^{(2)}$	$\surd^{(1)}$
	250	1601.62	$\surd^{(6)}$	$\surd^{(12)}$	$\surd^{(14)}$	$\surd^{(12)}$
0.6	100	1333.56	$\surd^{(14)}$	$\surd^{()}$	$\surd^{(3)}$	$\surd^{(1)}$
	150	1483.56	$\surd^{(11)}$	$\surd^{(11)}$	$\surd^{(6)}$	$\surd^{(4)}$
	200	1601.20	$\surd^{(5)}$	$\surd^{(6)}$	$\surd^{(3)}$	$\surd^{(3)}$
	250	1701.20	$\surd^{(6)}$	$\surd^{(10)}$	$\surd^{(8)}$	$\surd^{(2)}$
0.8	100	1458.83	$\surd^{(15)}$	$\surd^{(14)}$	$\surd^{(6)}$	$\surd^{(9)}$
	150	1594.08	$\surd^{(11)}$	$\surd^{(20)}$	$\surd^{(18)}$	$\surd^{(14)}$
	200	1690.57	$\surd^{(4)}$	$\surd^{(2)}$	$\surd^{(2)}$	$\surd^{(1)}$
	250	1740.57	$\surd^{(1)}$	$\surd^{(3)}$	$\surd^{(6)}$	$\surd^{(2)}$
1.0	100	1556.63	$\surd^{(16)}$	$\surd^{(19)}$	$\surd^{(13)}$	$\surd^{(7)}$
	150	1640.57	$\surd^{(3)}$	$\surd^{(4)}$	$\surd^{(6)}$	$\surd^{(5)}$
	200	1690.57	$\surd^{(2)}$	$\surd^{(3)}$	$\surd^{(3)}$	$\surd^{(4)}$
	250	1740.57	$\surd^{(3)}$	$\surd^{(10)}$	$\surd^{(8)}$	$\surd^{(11)}$

Appendix C

Appendix of Convergence Curves of SOGA

Figures C.1, C.2, C.3, and C.4, show the convergence curves of problem instances 25TT, 50TT, 100TT, and 200TT.

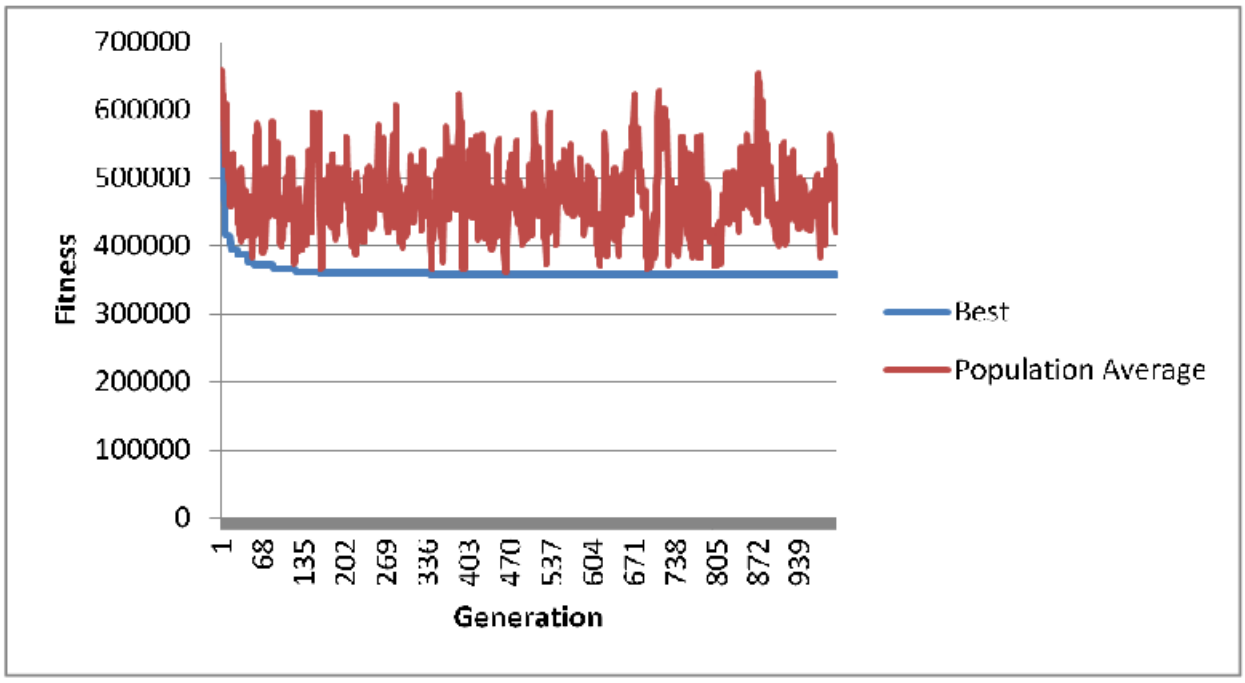

Figure C.1: Plot of Convergence Curve for Problem Instance 25TT

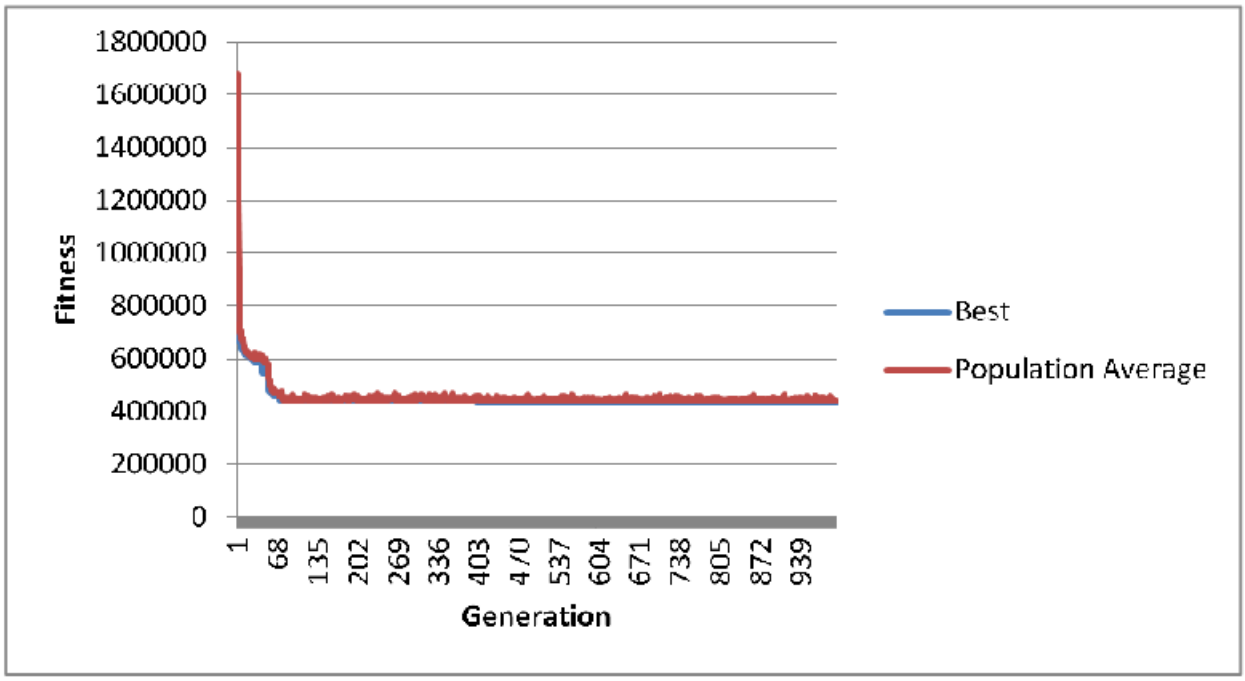

Figure C.2: Plot of Convergence Curve for Problem Instance 50TT

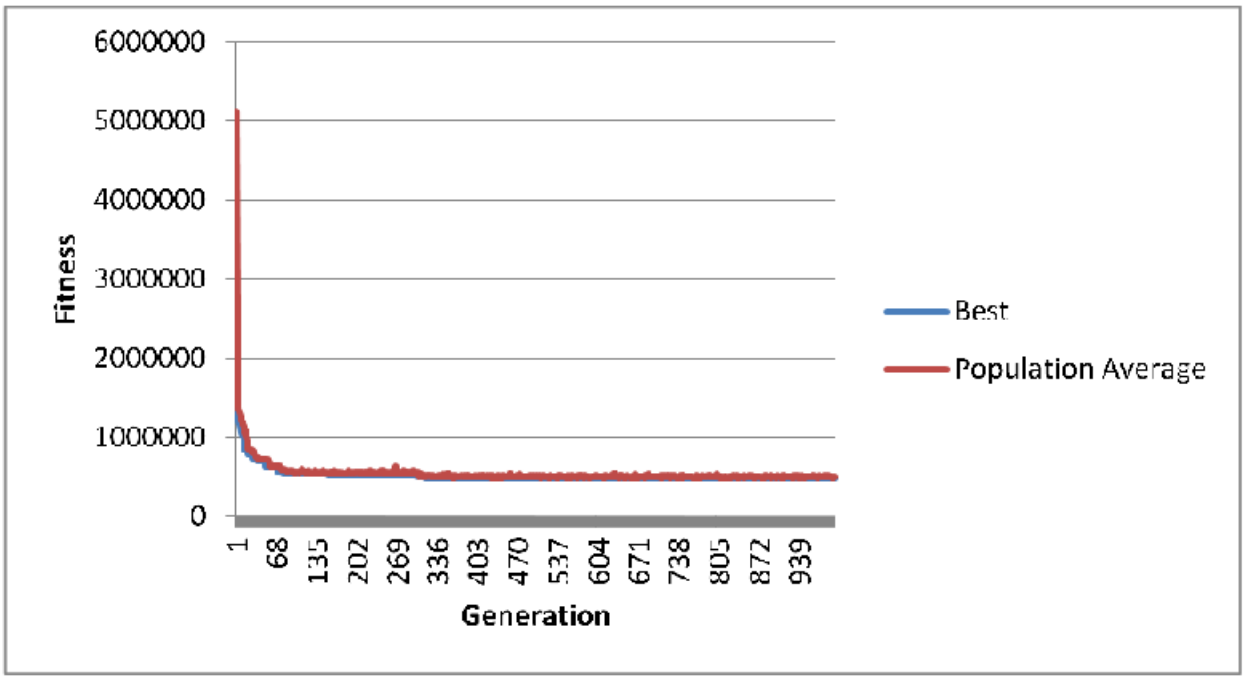

Figure C.3: Plot of Convergence Curve for Problem Instance 100TT

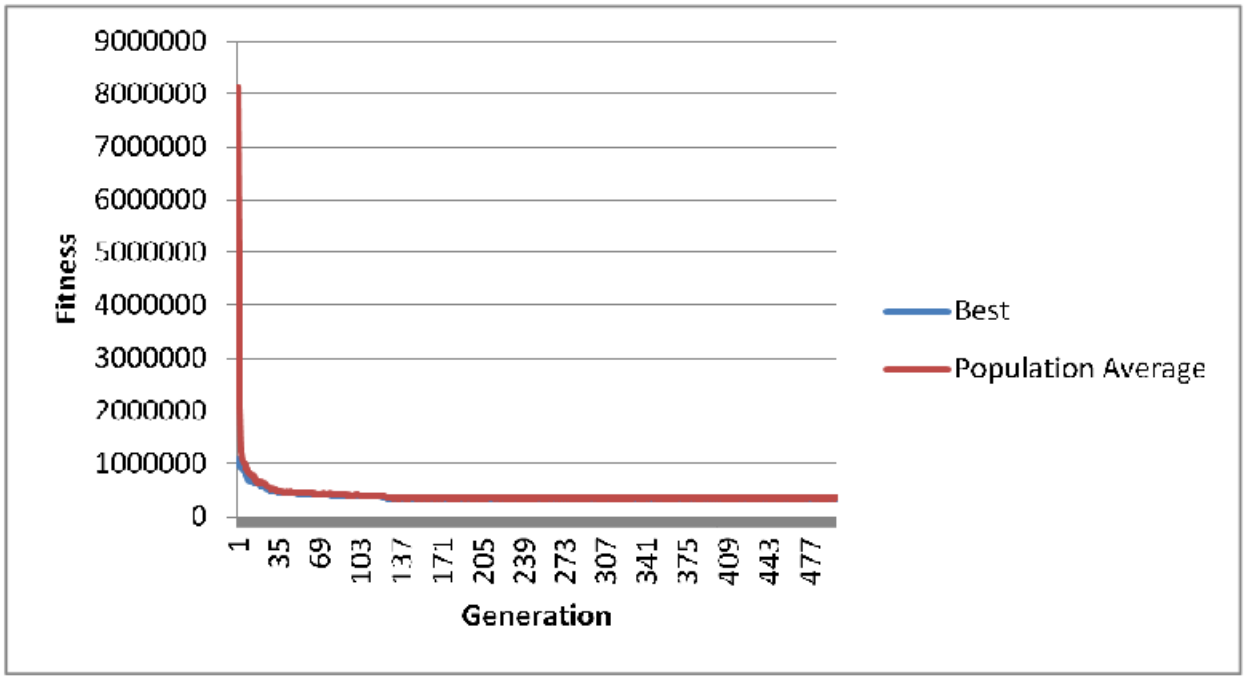

Figure C.4: Plot of Convergence Curve for Problem Instance 200TT